Confidence-based reasoning in stochastic constraint programming

Confidence-based Reasoning in Stochastic Constraint Programming[☆]

Roberto Rossi[a,1,*], Brahim Hnich[b], S. Armagan Tarim[c,3], Steven Prestwich[d,2]

[a]*Business School, University of Edinburgh, United Kingdom*
[b]*Department of Computer Engineering, Izmir University of Economics, Turkey*
[c]*Institute of Population Studies, Hacettepe University, Turkey*
[d]*Insight Centre for Data Analytics, University College Cork, Ireland*

Abstract

In this work we introduce a novel approach, based on sampling, for finding policies that are likely to be solutions to stochastic constraint satisfaction problems and constraint optimisation problems. Our approach reduces the size of the original problem being analysed and it guarantees that, with a given confidence probability, the policies produced by solving this reduced problem satisfy the chance constraints in the original model within prescribed error tolerance thresholds. To do so, we blend concepts from stochastic constraint programming and statistics. The strategy introduced can be immediately employed in concert with existing approaches for solving stochastic constraint programs. We illustrate our novel approach on a number of stochastic combinatorial optimisation problems. A thorough computational study demonstrates the effectiveness of our approach.

Keywords: stochastic constraint programming, sampled SCSP, (α,ϑ)-solution, confidence-based reasoning, confidence interval analysis, global chance constraint

1. Introduction

Solving large-scale problems that are stochastic in nature is a computationally hard task. To date, no general purpose method exists for solving this class of problems. Even trivial instances with a dozen of decisions to be made and the same number of random parameters to be considered typically require a computational effort out of reach even for the most advanced hardware/software combination.

[☆]This work is an extended version of [1]

[*]**Corresponding author**. Roberto Rossi, Business School, University of Edinburgh, 29 Buccleuch place, EH8 9JS, Edinburgh, UK. Tel. +44(0)131 6515239, Fax. +44 (0)131 650 8077

Email addresses: roberto.rossi@ed.ac.uk (Roberto Rossi), brahim.hnich@ieu.edu.tr (Brahim Hnich), armagan.tarim@hacettepe.edu.tr (S. Armagan Tarim), s.prestwich@4c.ucc.ie (Steven Prestwich)

[1]R. Rossi is supported by the University of Edinburgh CHSS Challenge Investment Fund

[2]This publication has emanated from research supported in part by a research grant from Science Foundation Ireland (SFI) under Grant Number SFI/12/RC/2289

[3]S. Armagan Tarim is supported by the Scientific and Technological Research Council of Turkey (TUBITAK) under grant no MAG-110M500.

We argue that, in solving large-scale problems that are stochastic in nature, one should not consider looking at the ultimate feasible/optimal solution; rather, the decision maker should aim for a solution that she "sufficiently trusts," which she may claim to be optimal or feasible with a given confidence level and for which a certain degree of error may be tolerated. In order to obtain such a solution, the decision maker should only look at a possibly limited number of samples drawn from the random variables in the model. In other words, she should try to "estimate" the quality of this solution.

In this work we introduce one such approach. Our approach has several analogies with established techniques in statistics. When a survey is conducted on a sample population — e.g. an electoral poll — a statistician typically associates a certain confidence level with the results obtained from the chosen sample population. For instance, one may claim that there is a 90% chance that the actual mean being estimated is within a given interval. We argue that the very same approach may be adopted in stochastic decision making. When a decision is to be made under uncertainty, one typically requires a number of constraints to be satisfied at a prescribed probability. For instance, if x is a decision variable and r is a random variable distributed according to some distribution law, we may require that $\Pr\{x \geq r\} \geq \beta$, which means we require the probability the constraint $x \geq r$ is satisfied by the assignment chosen for x to be greater or equal to a given threshold β. In general, in a stochastic constraint satisfaction problem, we may observe a number of constraints of the form $\Pr\{\langle\text{constraint}\rangle\} \geq \beta$, which we call chance constraints. If these constraints do not admit any closed form solution and are complex enough to rule out any chance of obtaining an exact solution, we suggest that — similarly to what is done in statistics — one may introduce a confidence level α and a tolerated estimation error $\pm\vartheta$. The decision maker, instead of looking for an exact solution, may then aim to "estimate" — according to the chosen α and ϑ — if the actual satisfaction probability guaranteed by an assignment is greater or equal to the given target value β for each of the chance constraints in the model. By choosing given values for α and ϑ the set of solutions may vary. For this reason we will introduce a new notion of solution that is parameterised by these two parameters and that we call (α, ϑ)-solution. Intuitively, as α tends to 1 and ϑ tends to 0 the set of (α, ϑ)-solutions will converge to the set of actual solutions to the original stochastic constraint satisfaction problem, which we therefore rename $(1, 0)$-solutions.

In this work, we make the following contributions to the stochastic constraint programming literature:

- we discuss how to obtain compact instances of complex stochastic constraint programs via sampling, we call these instances "sampled SCSPs;"
- we introduce the concepts of (α, ϑ)-solution and of (α, ϑ)-solution set;
- we show how the above tools can be employed in order to find approximate solutions to generic stochastic constraint satisfaction/optimisation problems;
- we conduct a thorough computational study on three well-known stochastic combinatorial problems.

This work is structured as follows: in Section 2 we introduce the relevant formal background in constraint programming and stochastic constraint programming; in Section 3 we discuss existing techniques for modeling and solving stochastic constraint programs; in Section 4 we introduce sampled SCSPs; in Section 5 we introduce the relevant formal background in confidence interval analysis; in Sections 6 and 7 we discuss properties of the solutions of sampled SCSPs

and formally introduce (α, ϑ)-solutions and (α, ϑ)-solution sets; in Section 8 we extend our discussion to stochastic constraint optimisation problems; in Section 9 we discuss connections with established techniques in statistics; in Section 10 we present our computational study; in Sections 11 and 12 we discuss related works and future research directions; finally, in Section 13 we draw conclusions.

2. Formal background

A Constraint Satisfaction Problem (CSP) [2] consists of a set of variables, each with a finite domain of values, and a set of constraints specifying allowed combinations of values for some variables. A *solution* to a CSP is an assignment of variables to values in their respective domains such that all of the constraints are satisfied. Constraint solvers typically explore partial assignments enforcing a local consistency property. A constraint c is *generalized arc consistent* (*GAC*) iff when a variable is assigned any of the values in its domain, there exist compatible values in the domains of all the other variables of c. In order to enforce a local consistency property on a constraint c during search, we employ filtering algorithms that remove inconsistent values from the domains of the variables of c. These filtering algorithms are repeatedly called until no more values are pruned. This process is called *constraint propagation*.

The following definitions are based on [3, 4]. An m-stage stochastic constraint satisfaction problem (SCSP) [5] is defined as a 7-tuple $\langle V, S, D, P, C, \beta, L\rangle$, where V is a set of decision variables and S is a set of random variables, D is a function mapping each element of V (respectively, S) to a domain (respectively, support) of potential values. In classical SCSPs both decision variable domains and random variable supports are assumed to be finite. P is a function mapping each element of S to a probability distribution for its associated support. C is a set of constraints over a non-empty subset of decision variables and a subset of random variables. If a constraint constrains only decision variables, then we call it a deterministic constraint; if it constrains both decision and random variables, then we call it a stochastic constraint. β is a function mapping each stochastic constraint $h \in C$ to β_h, which is a threshold value in the interval $(0, 1]$. If this threshold is strictly less than 1, then the stochastic constraint is a chance constraint. Note that it does not make sense to set $\beta_c < 1$ for a deterministic constraint. $L = [\langle V_1, S_1\rangle, \ldots, \langle V_i, S_i\rangle, \ldots, \langle V_m, S_m\rangle]$ is a list of *decision stages* such that each $V_i \subseteq V$, each $S_i \subseteq S$, the V_i form a partition of V, and the S_i form a partition of S.

To solve an m-stage SCSP an assignment to the variables in V_1 must be found such that, given random values for S_1, assignments can be found for V_2 such that, given random values for S_2, …, assignments can be found for V_m so that, given random values for S_m, the deterministic constraint are satisfied and the stochastic constraints are satisfied in the fraction of all possible scenarios specified by function β. Under the assumption that random variable supports are finite, the solution of an m-stage SCSP is, in general, represented by means of a *policy tree* [6]. The arcs in such a policy tree represent values observed for random variables whereas nodes at each level represent the decisions associated with the different stages. We call the policy tree of an m-stage SCSP that is a solution a *satisfying policy tree*.

Let $\mathcal{S}$ denote the space of policy trees that are solutions to a SCSP. We may be interested in finding a policy tree $s \in \mathcal{S}$ that maximizes the value of a given objective function $f(\cdot)$ over a subset of stochastic variables and a non-empty subset of decision variables. A stochastic constraint optimization problem (SCOP) is then defined in general as $\max_{s\in\mathcal{S}} f(s)$.

In order to simplify the presentation, we assume without loss of generality, that each $V_i = \{x_i\}$ and each $S_i = \{s_i\}$ are singleton sets. All the results can be easily extended in order to consider

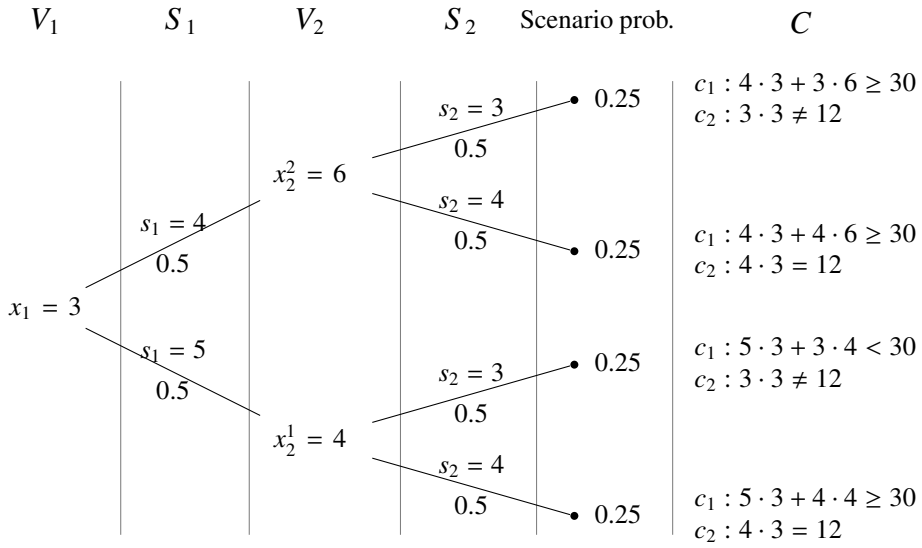

Figure 1: Policy tree for the SCSP in Example 1

$|V_i| > 1$ and $|S_i| > 1$ (see [4]). Intuitively, if S_i comprises more than one random variable, it is always possible to aggregate these variables into a single multivariate random variable [7] by convoluting them. If V_i comprises more than one decision variable, in the following discussion the term decision variable should be interpreted as a set of decision variables. Let $S = \{s_1, s_2, \ldots, s_m\}$ be the set of all random variables and $V = \{x_1, x_2, \ldots, x_m\}$ be the set of all decision variables.

Let p be a path from the root node of the policy tree to a leaf. Let Ψ denote the set of all distinct paths of a policy tree. For each $p \in \Psi$, we denote by arcs(p) the sequence of all the arcs in p whereas nodes(p) denotes the sequence of all nodes in p. We denote by $\Omega = \{\text{arcs}(p) | p \in \Psi\}$ the set of all scenarios of the policy tree. The probability of $\omega \in \Omega$ is given by $\Pr\{\omega\} = \prod_{i=1}^{m} \Pr\{s_i = \bar{s}_i\}$, where $\Pr\{s_i = \bar{s}_i\}$ is the probability that random variable s_i takes value $\bar{s}_i$.

Now consider a constraint $h \in C$ with a specified threshold level β_h. Consider a policy tree $\mathcal{T}$ for the SCSP and a path $p \in \mathcal{T}$. Let $h_{\downarrow p}$ be the deterministic constraint obtained by substituting the random variables in h with the corresponding values ($\bar{s}_i$) assigned to these random variables in arcs(p). Let $\bar{h}_{\downarrow p}$ be the resulting tuple obtained by substituting the decision variables in $h_{\downarrow p}$ by the values ($\bar{x}_i$) assigned to the corresponding decision variables in nodes(p). We say that h is *satisfied wrt to a given policy tree* $\mathcal{T}$ iff

$$\sum_{p \in \Psi : \bar{h}_{\downarrow p} \in h_{\downarrow p}} \Pr\{\text{arcs}(p)\} \geq \beta_h.$$

Definition 1. *Given an m-stage SCSP $\mathcal{P}$ and a policy tree $\mathcal{T}$, $\mathcal{T}$ is a satisfying policy tree to $\mathcal{P}$ iff every constraint of $\mathcal{P}$ is satisfied wrt $\mathcal{T}$.*

Example 1. *Let us consider the two-stage SCSP in Fig. 2, whose stage structure is $L = [\langle V_1, S_1\rangle, \langle V_2, S_2\rangle]$; $V_1 = \{x_1\}$ and $S_1 = \{s_1\}$, $V_2 = \{x_2\}$ and $S_2 = \{s_2\}$. Random variable s_1 may take two possible values, 5 and 4, each with probability 0.5; random variable s_2 may also take*

Constraints:
(1) $\Pr\{s_1x_1 + s_2x_2 \geq 30\} \geq 0.75$
(2) $\Pr\{s_2x_1 = 12\} \geq 0.5$

Decision variables:
$x_1 \in \{1, 2, 3, 4\}$ $x_2 \in \{3, 4, 5, 6\}$

Random variables:
$s_1 \in \{4(0.5), 5(0.5)\}$ $s_1 \in \{3(0.5), 4(0.5)\}$

Stage structure:
$V_1 = \{x_1\}, V_2 = \{x_2\}$ $S_1 = \{s_1\}, S_2 = \{s_2\}$
$L = [\langle V_1, S_1\rangle, \langle V_2, S_2\rangle]$

Figure 2: The two-stage SCSP in Example 1

two possible values, 3 and 4, each with probability 0.5. The domain of x_1 *is* $\{1, \ldots, 4\}$*, the domain of* x_2 *is* $\{3, \ldots, 6\}$*. There are two chance constraints*[4] *in C,* $c_1 : \Pr\{s_1x_1 + s_2x_2 \geq 30\} \geq 0.75$ *and* $c_2 : \Pr\{s_2x_1 = 12\} \geq 0.5$*. In this case, the decision variable* x_1 *must be set to a unique value before random variables are observed, while decision variable* x_2 *takes a value that depends on the observed value of the random variable* s_1*. A possible solution to this SCSP is the satisfying policy tree shown in Fig. 1 in which* $x_1 = 3, x_2^1 = 4$ *and* $x_2^2 = 6$*, where* x_2^1 *is the value assigned to decision variable* x_2*, if random variable* s_1 *takes value* 5*, and* x_2^2 *is the value assigned to decision variable* x_2*, if random variable* s_1 *takes value* 4*.*

As example 1 shows, a solution to a SCSP is not simply an assignment of the decision variables in V to values, but it is instead a satisfying policy tree.

3. Existing approaches for modeling and solving SCSPs

In [6], the authors discuss an equivalent scenario-based reformulation for SCSPs. This reformulation makes it possible to compile SCSPs down into conventional (non-stochastic) CSPs. For example, the multi-stage SCSP described in example 1 is compiled down to its deterministic equivalent CSP shown in Fig. 3. The decision variables x_1^1, x_2^1, and x_2^2 represent the nodes of the policy tree. The variable x_1 is decided at stage 1 so we have one copy of it (x_1^1) whereas since x_2 is to be decided at stage 2 and since s_1 has two values, we need two copies for x_2, namely x_2^1 and x_2^2. Chance constraint c_1 is compiled down into constraints (1),...,(5), whilst chance constraint c_2 is compiled down into constraints (6),...,(10). Constraints (1),...,(4) are reification constraints in which every binary decision variable $Z_{c_1}^{\omega}$ is 1 iff in scenario $\omega \in \{1, \ldots, 4\}$ constraint $\bar{s}_1x_1^1 + \bar{s}_2x_2^i \geq 30$ — where $i \in \{1, 2\}$ identifies the copy of decision variable x_2 associated with scenario ω — is satisfied. Finally, constraint (5) enforces that the satisfaction probability achieved

[4] In what follows, for convenience, we may denote a chance constraint by using the notation "$\Pr\{\langle cons\rangle\} \geq \beta$", meaning that constraint $\langle cons\rangle$, constraining decision and random variables, should be satisfied with probability greater or equal to β.

Constraints:

(1) $(5x_1^1 + 4x_2^1 \geq 30) \leftrightarrow (Z_{c_1}^1 = 1)$ (6) $(4x_1^1 = 12) \leftrightarrow (Z_{c_2}^1 = 1)$

(2) $(5x_1^1 + 3x_2^1 \geq 30) \leftrightarrow (Z_{c_1}^2 = 1)$ (7) $(3x_1^1 = 12) \leftrightarrow (Z_{c_2}^2 = 1)$

(3) $(4x_1^1 + 4x_2^2 \geq 30) \leftrightarrow (Z_{c_1}^3 = 1)$ (8) $(4x_1^1 = 12) \leftrightarrow (Z_{c_2}^3 = 1)$

(4) $(4x_1^1 + 3x_2^2 \geq 30) \leftrightarrow (Z_{c_1}^4 = 1)$ (9) $(3x_1^1 = 12) \leftrightarrow (Z_{c_2}^4 = 1)$

(5) $\sum_{\omega=1}^{4} 0.25 Z_{c_1}^{\omega} \geq \beta_{c_1}$ (10) $\sum_{\omega=1}^{4} 0.25 Z_{c_2}^{\omega} \geq \beta_{c_2}$

Decision variables:

$x_1 \in \{1, 2, 3, 4\}$,
$x_2^1 \in \{3, 4, 5, 6\}$,
$x_2^2 \in \{3, 4, 5, 6\}$,
$Z_h^{\omega} \in \{0, 1\}, \quad \forall \omega = 1, \ldots, 4, \quad \forall h \in \{c_1, c_2\}.$

Figure 3: Deterministic equivalent CSP for example 1

must be greater or equal to the required threshold $\beta_{c_1} = 0.75$. A similar reasoning applies to constraints (6), . . . ,(10). The scenario-based reformulation approach allows us to exploit the full power of existing constraint solvers. However, as pointed out in [4], it has a number of serious drawbacks that might prevent it from being applied in practice: weakened constraint propagation and increased space requirements. The authors in [4] therefore proposed an alternative approach that overcomes these drawbacks. More specifically, they proposed a general purpose approach for filtering global chance constraints. Global chance constraints were introduced first in [8] and bring together the reasoning power of global constraints from constraint programming and the expressive power of chance constraints from stochastic programming. The approach in [4] is able to reuse existing propagators available for the respective deterministic global constraint which corresponds to a given global chance constraint when all the random variables are replaced by constant parameters.

Unfortunately, both the above approaches operate under the assumption that the number of scenarios must be finite, otherwise a solution cannot be expressed as a finite number of possible decisions. This, in turn, means that complete approaches such as the one in [6] and in [4] can only deal with stochastic variables having finite supports. Furthermore, these approaches do not scale well, since even problems having a limited number of stochastic variables with large support immediately produce policy trees whose size makes impractical the use of a complete method.

In practice, it is often the case that random variables either range over continuous supports or have a very large number of possible values in their domain. In [6], the authors therefore proposed to employ a number sampling strategies in order to reduce a-priori the support of stochastic variables and therefore produce SCSPs that are manageable. Unfortunately, this strategy is purely heuristic and does not provide any guarantee to the decision maker that a given assignment is, in fact, a solution. Other heuristic approaches such as the one in [9] have been also proposed. In this approach, a neural network is employed in order to encode a policy function that takes the best possible decision with respect to the past history of decisions taken and values observed for the stochastic variables. On the other hand, the modularity Constraint Programming, in which a number of different constraints can be aggregated in different ways so that a number of different problems can be solved by reusing existing filtering algorithms, is partially lost in this technique.

For this reason, in this work we argue that an alternative way of dealing with large SCSPs is to exploit sampling in order to "estimate" if a given assignment is consistent or not with respect to a given chance constraint. The "quality" of this estimate is determined by confidence interval analysis. Therefore, in contrast to [6], we do provide guarantees for the solutions found. In practice, we will explicitly state a confidence probability that constrains the actual number of possible mistakes in this estimation. In order to discuss our approach we will firstly introduce the concept of "sampled SCSP".

4. Sampled SCSPs

Consider a SCSP $\mathcal{P}$ over a set S of stochastic variables. Assume that stochastic variables are defined on supports comprising a large number of values. Solving the original SCSP clearly poses a hard combinatorial challenge, in fact the policy tree comprises a number of scenarios that is exponential in the size of stochastic variable domains. In this section we discuss how to *sample* a more compact SCSP, which comprises at most N scenarios, out of the original problem. We shall call this new problem $\widehat{\mathcal{P}}_N$ or "sampled SCSP" over N scenarios. Intuitively, a sampled SCSP is a reduced version of the original problem the solution of which is a policy tree that comprises a bounded number of paths sampled out of the original policy tree. In the following sections we will discuss under which conditions the solution to a sampled SCSP $\widehat{\mathcal{P}}_N$ is, with a certain confidence probability, likely to be also a solution to the original SCSP $\mathcal{P}$.

We shall here discuss how to employ Simple Random Sampling to obtain a sampled SCSP out of the original problem. Of course, more advanced stratified sampling techniques may be used in order to reduce variance and improve the effectiveness of the approach. Nevertheless, due to large number of topics already covered in this work, we leave this discussion as future work.

Consider a complete realization, $\bar{s}_1, \ldots, \bar{s}_m$, for the stochastic variables in S obtained by sampling a value from the support $D(s_i)$ of each of the stochastic variables $s_i \in S$ according to its probability distribution $P(s_i)$. From the definition of policy tree it is clear that there always exists a path associated with this realization. In other words, this realization corresponds to one of the scenarios comprised in the policy tree.

Consider a policy tree $\mathcal{T}$ for $\mathcal{P}$ and N complete sets of random variable realizations generated independently:

$$\{\bar{s}_1^1, \ldots, \bar{s}_m^1\}, \{\bar{s}_1^2, \ldots, \bar{s}_m^2\}, \ldots, \{\bar{s}_1^N, \ldots, \bar{s}_m^N\},$$

where $\bar{s}_j^i$ is the realized value for random variable j observed in the i-th set of realizations. We remove from $\mathcal{T}$ every path which corresponds to an arc labeling not observed in the former N complete realizations. We call the resulting reduced policy tree $\widehat{\mathcal{T}}$.

Let $\widehat{\Psi}$ denotes the reduced set of distinct paths in $\widehat{\mathcal{T}}$. The probability of each of the remaining path $p \in \widehat{\Psi}$, i.e. $\Pr\{\text{arcs}(p)\}$, is simply set equal to the frequency of occurrence of such a path in the above N realizations. Of course, $\widehat{\mathcal{T}}$ represents a policy tree for a different SCSP than the one we started with. We call this new problem the sampled SCSP $\widehat{\mathcal{P}}_N$.

Now consider a chance constraint $h \in C$ with a specified threshold level β_h, a policy tree $\widehat{\mathcal{T}}$ for the sampled SCSP $\widehat{\mathcal{P}}_N$ and a path $p \in \mathcal{T}$. We say that h is *satisfied wrt to a given policy tree* $\widehat{\mathcal{T}}$ iff

$$\sum_{p \in \widehat{\Psi} : \bar{h}_{\downarrow p} \in h_{\downarrow p}} \Pr\{\text{arcs}(p)\} \geq \beta_h.$$

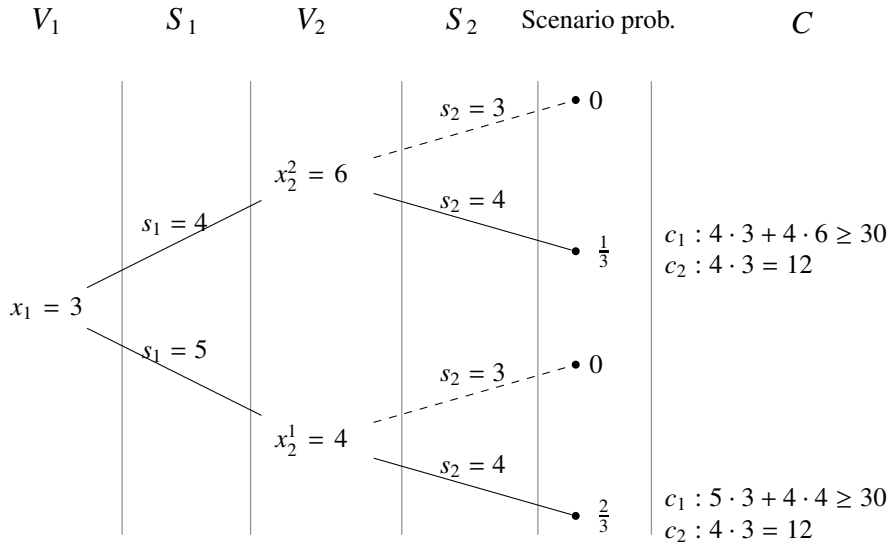

Figure 4: Policy tree for the sampled SCSP in example 2

Example 2. *Let us consider the two-stage SCSP $\mathcal{P}$ discussed in example 1. We set $N = 3$ and we derive a sampled SCSP $\widehat{\mathcal{P}}_N$. By using simple random sampling we draw the following three complete realizations for random variables in $\mathcal{P}$:*

$$\{\bar{s}_1^1 = 5, \bar{s}_2^1 = 4\}, \{\bar{s}_1^2 = 4, \bar{s}_2^2 = 4\}, \{\bar{s}_1^3 = 5, \bar{s}_2^3 = 4\}.$$

A possible solution to the sampled SCSP $\widehat{\mathcal{P}}_N$ is the satisfying policy tree shown in Fig. 4, in which $x_1 = 3, x_2^1 = 4$ and $x_2^2 = 6$, where x_2^1 is the value assigned to decision variable x_2, if stochastic variable s_1 takes value 5, and x_2^2 is the value assigned to decision variable x_2, if stochastic variable s_1 takes value 4. The above policy tree has two paths sampled out of the original tree: p_1 has an associated probability of 2/3, since we observed two occurrences of the scenario associated with this path over the 3 complete realisations sampled for the random variables; p_2 has an associated probability of 1/3, since we observed a single occurrence of the scenario associated with this path over the 3 complete realisations sampled for the random variables. Paths that were not observed in the sampled realisations have an associated probability equal to zero and are not considered.

It should be noted that every policy tree $\widehat{\mathcal{T}}$ for a sampled SCSP $\widehat{\mathcal{P}}$ can be employed as a (partial) policy tree for the original SCSP $\mathcal{P}$. Nevertheless, by sampling we lose completeness. If at stage i in $\mathcal{P}$ we observe, for a given random variable, a realised value that is not comprised in $\widehat{\mathcal{T}}$, it will be of course impossible to determine the correct decisions for subsequent stages. By taking a conservative point of view, this means that all paths in the corresponding subtree will never be satisfied. In multi-stage SCSPs, and especially in those including random variables with continuous support, this prevents the direct use of the approach that will be discussed in this work. In this case, it is therefore essential to adopt a "rolling horizon" approach [10] in order to reduce the original multi-stage SCSPs to a sequence of multi-stage sampled SCSPs. Under this strategy, our aim is to fix decisions at stage one, and make sure that compatible values exist for decision

variables that appear, for subsequent stages, in $\widehat{\mathcal{T}}$. Future decisions are not fixed because, after observing the realised values for random variables at stage one, the problem is solved again by taking into account new available information; decision variables that were previously associated with stage two "slide" and become stage one decisions. The original problem is thus reduced to a sequence of multi-stage sampled SCSPs. We will apply this technique to handle the two-stage problem discussed in Section 10.2: the stochastic multiprocessor scheduling problem with release time and deadlines.

In general, even if our problem is multi-stage, it is still possible that the remaining paths form a (partial) policy tree that is a satisfying policy tree for $\mathcal{P}$. In Example 2, incidentally, the satisfying policy tree for the sampled SCSP is also a satisfying policy tree for the original 2-stage SCSP. It is relatively intuitive to see that if we repeatedly produce new sampled SCSPs with $N = 3$, with a certain probability a satisfying policy tree for the sampled SCSP will also be a satisfying policy tree for the original SCSP within the given error tolerance threshold ϑ for the satisfaction probability β_c of each chance constraint c in the model. The rest of this work is mainly concerned with the estimation of this probability. We next introduce the relevant background in confidence interval analysis, the key tool we employ to perform this estimation.

5. Confidence interval analysis

Confidence interval analysis is a well established technique in statistics. Informally, confidence intervals are a useful tool for computing, from a given set of experimental results, a range of values that, with a certain confidence level (or confidence probability), will cover the actual value of a parameter that is being estimated.

Consider a discrete random variable that follows a Bernoulli distribution. Accordingly, such a variable may produce only two outcomes, i.e. "yes" and "no", with probability q and $1 - q$, respectively.

Let us assume that the value q — the "yes" probability — is unknown. Obviously, if we observe the outcome of a Bernoulli trial once, the data collected will not reveal much about the value of q. Nevertheless, in practice, we may be interested in "estimating" q, by repeatedly observing the behavior of the random variable in a sequence of Bernoulli trials. This problem is well-known in statistics and both exact and approximate techniques are available for performing this estimation [11, 12]. The estimation produced by the methods available in the literature typically does not come as a point estimate, rather it consists of an interval of values computed from a set of representative samples for the quantity being estimated. This interval is known as "confidence interval" and consists of a range of values that, with a certain confidence probability α, covers the actual value of the parameter that is being estimated.

A method that is commonly classified as the "exact confidence intervals" for the Binomial distribution has been introduced by Clopper and Pearson in [11]. This method uses the Binomial cumulative distribution function (CDF) in order to build the interval from the data observed. Clopper-Pearson interval is a symmetric two-sided confidence interval. It can be however also expressed as a single-sided interval. Clopper-Pearson single-sided intervals can be written as $(p_{lb}, 1)$ and $(0, p_{ub})$ where

$$p_{lb} = \min\{q | \Pr\{\text{bin}(N; q) \geq X\} \geq 1 - \alpha\},$$
$$p_{ub} = \max\{q | \Pr\{\text{bin}(N; q) \leq X\} \geq 1 - \alpha\},$$

X is the number of successes (or "yes" events) observed in the sample, $\text{bin}(N; q)$ is a binomial

random variable with N trials and probability of success q and α is the confidence probability. Note that we assume $p_{lb} = 0$ when $X = 0$ and that $p_{ub} = 1$ when $X = N$.

Because of the close relationship between Binomial distribution and the Beta distribution, the Clopper-Pearson interval is sometimes presented in an alternative format that uses percentiles from the beta distribution [13]:

$$p_{lb} = 1 - \text{beta}^{-1}(\alpha, N - X + 1, X),$$
$$p_{ub} = 1 - \text{beta}^{-1}(1 - \alpha, N - X, X + 1),$$

where beta^{-1} denotes the inverse Beta distribution. This form can be efficiently evaluated by existing algorithms.

An interesting property of confidence intervals related to the estimation of the "success" probability associated with a Bernoulli trial consists in the fact that, given a confidence probability, it is possible to derive mathematically, by performing a worst case analysis, the minimum number of samples that should be observed in order to produce a confidence interval of a given size.

Therefore, for a given confidence probability α, it is possible to determine the minimum number of samples that should be considered in order to achieve a margin of error of $\pm\vartheta$ in the estimation of the "success" probability of a Bernoulli trial. This computation plays a central role in our novel approach. In fact, intuitively estimating the satisfaction probability of a chance constraint is equivalent to estimating the "success" probability of the associated Bernoulli trial.

6. (α,ϑ) solutions

We will now characterize the probability that the solution of a sampled SCSPs $\widehat{\mathcal{P}}_N$ over N scenarios, which may be computed by using any of the existing approaches discussed in Section 3, is a solution to the original *single-stage* SCSP $\mathcal{P}$. As discussed, these results are also applicable to multi-stage problems, provided that a rolling horizon approach is adopted.

We will firstly discuss what the minimum value for N is in order to achieve a predefined probability α that a given policy tree $\mathcal{T}$ that satisfies a chance constraint h in the sampled SCSPs $\widehat{\mathcal{P}}_N$ also satisfies the same chance constraint in the original SCSP $\mathcal{P}$. Since a policy tree $\mathcal{T}$ in $\widehat{\mathcal{P}}_N$ by definition only comprises a subset $\widehat{\Psi}$ of all the paths that constitute a policy tree for the original SCSP $\mathcal{P}$, this policy tree, in order to satisfy h in the original SCSP $\mathcal{P}$, must clearly provide a sufficient satisfaction probability regardless of the scenarios that have been ignored by the sampling process.

Consider a confidence probability α and a margin of error of $\pm\vartheta$; The number of scenarios N for the sampled SCSP depends on ϑ, α and also β, which we recall is the satisfaction probability we aim for our chance constraint h.

Definition 2. *N is computed as the minimum value for which*

$$\max(p^{\beta}_{ub} - \beta, \beta - p^{\beta}_{lb}) \leq \vartheta,$$

where $(p^{\beta}_{lb}, p^{\beta}_{ub})$, is the Clopper-Pearson confidence interval for a confidence probability α, and round(βN) "successes" in N trials; round() approximates the value to the nearest integer.[5]

[5]This is justified by the fact that the Clopper-Pearson interval is, in fact, a step function — see [11], p. 405 — since the Binomial is a discrete probability distribution.

Definition 3. *Any policy tree $\mathcal{T}$, which can be proved to satisfy h in $\mathcal{P}$ with probability α, satisfies h in $\mathcal{P}$ with probability α if it satisfies h in $\widehat{\mathcal{P}}_N$. Conversely, any policy tree $\mathcal{T}$, which can be proved to not satisfy h in $\mathcal{P}$ with probability α, does not satisfy h in $\mathcal{P}$ with probability α, if it does not satisfy h in $\widehat{\mathcal{P}}_N$.*

Proposition 1. *A policy tree $\mathcal{T}$ can be proved to satisfy h in $\mathcal{P}$ with probability α if the actual satisfaction probability $\delta > \beta$ provided by $\mathcal{T}$ wrt h is such that $\delta \geq p^{\beta}_{ub}$. Conversely, if the actual satisfaction probability $\delta < \beta$ provided by $\mathcal{T}$ wrt h is such that $\delta \leq p^{\beta}_{lb}$ $\mathcal{T}$ can be proved to not satisfy h in $\mathcal{P}$ with probability α.*

Proof. Let $\delta \geq p^{\beta}_{ub}$. By definition,

$$p^{\beta}_{ub} = \max\{q | \Pr\{\text{bin}(N; q) \leq \text{round}(\beta N)\} \geq 1 - \alpha.$$

Therefore, it is clear that $\Pr\{\text{bin}(N; \delta) \leq \text{round}(\beta N)\} < 1 - \alpha$. This means that

$$\Pr\left\{ \sum_{p \in \widehat{\Psi}: \bar{h}_{\downarrow p} \in h_{\downarrow p}} \Pr\{\text{arcs}(p)\} \leq \beta \right\} < 1 - \alpha,$$

where we recall that $\widehat{\Psi}$ is the set of paths in the sampled SCSP $\widehat{\mathcal{P}}_N$. This implies

$$\Pr\left\{ \sum_{p \in \widehat{\Psi}: \bar{h}_{\downarrow p} \in h_{\downarrow p}} \Pr\{\text{arcs}(p)\} \geq \beta \right\} \geq \alpha.$$

Therefore, by using the test

$$\sum_{p \in \widehat{\Psi}: \bar{h}_{\downarrow p} \in h_{\downarrow p}} \Pr\{\text{arcs}(p)\} \geq \beta,$$

a policy tree $\mathcal{T}$ can be proved to satisfy h in $\mathcal{P}$ with probability α. Conversely, let $\delta \leq p^{\beta}_{lb}$. By definition,

$$p^{\beta}_{lb} = \min\{q | \Pr\{\text{bin}(N; q) \geq \text{round}(\beta N)\} \geq 1 - \alpha.$$

Therefore, it is clear that

$$\Pr\{\text{bin}(N; \delta) \geq \text{round}(\beta N)\} < 1 - \alpha.$$

This means that

$$\Pr\left\{ \sum_{p \in \widehat{\Psi}: \bar{h}_{\downarrow p} \in h_{\downarrow p}} \Pr\{\text{arcs}(p)\} \geq \beta \right\} < 1 - \alpha,$$

which implies

$$\Pr\left\{ \sum_{p \in \widehat{\Psi}: \bar{h}_{\downarrow p} \in h_{\downarrow p}} \Pr\{\text{arcs}(p)\} \leq \beta \right\} \geq \alpha.$$

Therefore, by using the test

$$\sum_{p \in \widehat{\Psi}: \bar{h}_{\downarrow p} \in h_{\downarrow p}} \Pr\{\text{arcs}(p)\} \leq \beta,$$

a policy tree $\mathcal{T}$ can be proved to not satisfy h in $\mathcal{P}$ with probability α. □

Proposition 2. *Any policy tree $\mathcal{T}$ which provides a satisfaction probability $\delta \geq \beta + \vartheta$ wrt h in $\mathcal{P}$ can be proved to satisfy h in $\mathcal{P}$ with probability α. Any policy tree $\mathcal{T}$ which provides a satisfaction probability $\delta \leq \beta - \vartheta$ wrt h in $\mathcal{P}$ can be proved to not satisfy h in $\mathcal{P}$ with probability α.*

Proof. this directly follows from Definition 2 and Proposition 1. □

Proposition 3. *Any policy tree $\mathcal{T}$ which can not be proved to satisfy or to not satisfy h in $\mathcal{P}$ with probability α, can be either proved to satisfy h in $\mathcal{P}$ with probability γ, where γ is a probability ranging in $(0.5, \alpha($, if it satisfies h in $\widehat{\mathcal{P}}_N$, or to not satisfy h in $\mathcal{P}$ with probability γ, where γ is a probability ranging in $(0.5, \alpha($, if it does not satisfies h in $\widehat{\mathcal{P}}_N$.*

Proof. Consider the two limiting cases. (i) The actual satisfaction probability δ provided by $\mathcal{T}$ wrt h in $\mathcal{P}$ is exactly equal to β. Since the sample mean, used to estimate the satisfaction probability out of the N samples considered, is an unbiased estimator of δ, it will overestimate β with probability 0.5 and, similarly, it will underestimate β with probability 0.5; this sets the lower bound for γ. (ii) The actual satisfaction probability δ provided by $\mathcal{T}$ wrt h in $\mathcal{P}$ is exactly equal to $\beta + \vartheta$. From the proof of Proposition 1 it immediately follows that, in this case, $\gamma = \alpha$, and also that, if $\delta < \beta + \vartheta$ then $\gamma < \alpha$; this sets the upper bound for γ. □

Definition 4. *An (α, ϑ)-solution to a SCSP $\mathcal{P}$ is a policy tree $\widehat{\mathcal{T}}$ that at least with probability α provides for every chance constraint h_i in $\mathcal{P}$ with satisfaction threshold β_i a satisfaction probability greater or equal to $\beta_i - \vartheta$.*

It is apparent that ϑ may be interpreted as a parameter that the user can set in order to define a "region of indifference", i.e. $\beta \pm \vartheta$, for the satisfaction probability. In such a region, we assume that assignments can be safely misclassified with probability greater than α and that satisfaction probabilities remain in an acceptable range.

Example 3. *Consider the single-stage SCSP $\mathcal{P} = \langle V, S, D, P, C, \beta, L \rangle$, where $V = \{X_1, X_2\}$, $S = \{r_1, r_2\}$, $D(X_1) = D(X_2) = \{0, 1\}$, $D(r_1) = (0, 100)$, $P(r_1) = Uniform(0, 100)$, $D(r_2) = (0, 300)$, $P(r_2) = Uniform(0, 300)$, $C = \{c : C_1 \geq X_1 r_1 + X_2 r_2\}$, $\beta_c = 0.5$, and $L = [\langle V, S \rangle]$. $C_1 = 185$ is a constant. This problem comprises random variables defined on a continuous support and it cannot be solved by existing complete approaches to SCSPs. If we set $\alpha = 0.95$ and $\vartheta = 0.05$, from Definition 2 we compute the number of samples $N = 290$ required to guarantee that any solution to the sampled SCSP $\widehat{\mathcal{P}}$ over N samples is an (α, ϑ)-solution for $\mathcal{P}$.*

Furthermore, the simple structure of the constraint c considered in $\mathcal{P}$ allows us to perform some further analysis. Consider the assignment $X_1 = 1$ and $X_2 = 1$. A simple reasoning on the convolution of two independently non-identically distributed uniform random variables (see [14]) immediately suggests that this assignment is indeed inconsistent. r_1 and r_2 are two independently non-identically distributed uniform random variables. The distribution that results from their convolution is shown in Fig. 5. This distribution is shaped like a trapezoid. Clearly, since the area for the whole figure must be equal to 1, the area of each of the two rectangle triangles at the side of the trapezoid must be equal to 1/6. Consequently, the area of the internal rectangle must be equal to 2/3. It is easy to see that the cumulative distribution function for value 200 returns a probability of 0.5. Then, since 1/3(15/100)=0.05, the 0.45 quantile of the inverse cumulative distribution function which results from convoluting r_1 and r_2 is exactly equal to $C_1 = 185$. Therefore, since the satisfaction probability provided by the assignment $X_1 = 1$ and $X_2 = 1$ is equal to $\beta_c - \vartheta = 0.45$ (Fig. 6), this assignment will be correctly classified as inconsistent with probability α, when the sample size is set to $N = 290$.*

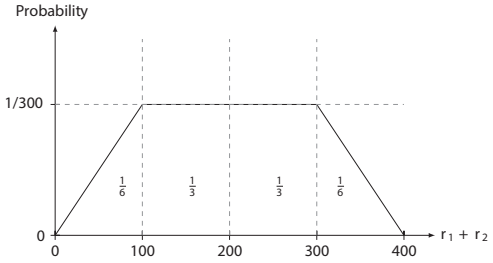

Figure 5: Probability density function of the convolution of two independently non-identically distributed uniform random variables r_1 and r_2.

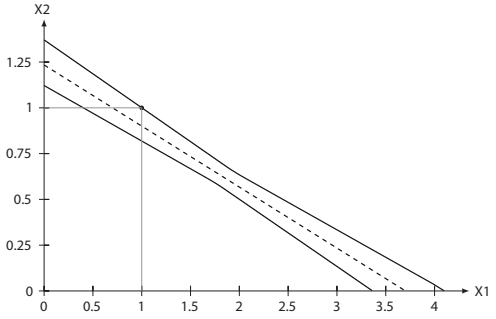

Figure 6: Feasible region for the SCSP in Example 1; the dashed line denotes the true boundary of constraint c. The upper solid line demarcates the set of solutions providing a satisfaction probability of at least $\beta - \vartheta$; the lower solid line demarcate the set of solutions providing a satisfaction probability of at least $\beta + \vartheta$.

Let $h_1, \ldots, h_k$ be k chance constraints in a SCSP $\mathcal{P}$. Let $\widehat{\mathcal{P}}$ be a sampled SCSP over N samples, where N is the number of samples required to guarantee a confidence level α and an error tolerance threshold ϑ for each constraint h_i considered independently, according to Definition 2.

Proposition 4. *Let $\widehat{\mathcal{T}}$ be a policy tree that is a solution to $\widehat{\mathcal{P}}$. Then $\widehat{\mathcal{T}}$ is an (α, ϑ)-solution for $\mathcal{P}$.*

Proof. Consider a chance constraint h_i. Let β_i be the respective satisfaction threshold. By definition, the probability that a solution $\widehat{\mathcal{T}}$ to $\widehat{\mathcal{P}}$ provides a service level less or equal to $\beta_i - \vartheta$ for h_i in $\mathcal{P}$ is less or equal to $1 - \alpha$. Therefore $\widehat{\mathcal{T}}$ is an (α, ϑ)-solution. Now consider a pair of chance constraints $\langle h_i, h_j \rangle$ with satisfaction thresholds β_i, β_j, respectively. The probability that a solution $\widehat{\mathcal{T}}$ to $\widehat{\mathcal{P}}$ provides a service level less or equal to $\beta_i - \vartheta$ for h_i and to $\beta_j - \vartheta$ for h_j in $\mathcal{P}$ is less or equal to $(1 - \alpha)^2$, in fact we must misclassify both the constraints in order to accept such a solution. Even a single constraint correctly classified will make $\widehat{\mathcal{T}}$ inconsistent w.r.t. $\widehat{\mathcal{P}}$. This reasoning can be easily generalized to k chance constraints, for which the probability becomes $(1 - \alpha)^k$. Noting that $(1 - \alpha)^k < \ldots < (1 - \alpha)^2 < (1 - \alpha)$ and that $1 - (1 - \alpha)^k \geq \alpha$ the probability that a solution is misclassified in a model comprising a single constraint, i.e. $(1 - \alpha)$, represents an upper bound for the probability that a solution $\widehat{\mathcal{T}}$ to $\widehat{\mathcal{P}}$ does not provide a satisfaction probability within the required tolerance threshold for one or more constraints in a generic model $\mathcal{P}$. By rephrasing, the probability that a solution $\widehat{\mathcal{T}}$ provides a satisfaction probability greater or equal to $\beta_i - \vartheta$ for each constraint h_i is greater or equal to α. Hence, by Definition 4, $\widehat{\mathcal{T}}$ is an (α, ϑ)-solution for $\mathcal{P}$. □

7. (α,ϑ) solution set

Consider policy tree $\mathcal{T}$, chance constraint h, and the associated indicator random variable

$$\tau = \begin{cases} 1 & \sum_{p \in \widehat{\Psi}: \bar{h}_{\downarrow p} \in h_{\downarrow p}} \Pr\{\text{arcs}(p)\} \geq \beta \\ 0 & \text{otherwise} \end{cases}$$

representing the test discussed in proposition 3. If the actual satisfaction probability δ provided by a policy tree $\mathcal{T}$ with respect to constraint h in $\mathcal{P}$ is exactly equal to β — i.e. h is binding

— τ takes value 1, i.e. it recognises $\mathcal{T}$ as feasible, with probability 0.5 and variance 0.25. The expected value of τ represents an unbiased and consistent estimator of the feasibility of $\mathcal{T}$. This means that τ can be used to identify an estimated boundary of constraint h that is unbiased and consistent. Furthermore, if the actual satisfaction probability δ provided by $\mathcal{T}$ with respect to h in $\mathcal{P}$ is exactly equal to $\beta - \vartheta$, τ takes value 1 with probability $(1 - \alpha)$.

Consider now T policy trees $\mathcal{T}_1, \ldots, \mathcal{T}_T$ for which the actual satisfaction probability δ wrt h in $\mathcal{P}$ is exactly equal to $\beta - \vartheta$; and associated random variables $\tau_1 \ldots, \tau_T$, each of which as discussed must take value 1 with probability $(1 - \alpha)$. Although we have fully characterised the probability distribution of a test τ_i involving a single policy tree $\mathcal{T}_i$, we have not characterised yet the conditional probability among tests carried out on a set of policy trees.

To motivate the following discussion, we refer once more to example 3. Assume that $D(X_1) = D(X_2) = (0, 5)$; i.e. decision variables are defined on continuos domains spanning from 0 to 5. Assignments $(X_1 = 4.1, X_2 = 0)$ and $(X_1 = 0, X_2 = 1.37)$ lie on the upper solid line shown in Fig. 6. Each of these two assignments provides a satisfaction probability of exactly $\beta - \vartheta$ with respect to constraint c in the original problem $\mathcal{P}$. From the discussion in Section 6 it follows that each of these two assignments is recognised as infeasible with probability α if $N = 290$. However, since r_1 and r_1 are independent (and therefore orthogonal) the probability that these two assignments are **both** recognised as infeasible is only α^2. We next discuss how to address the issue of correctly classifying multiple policy trees according to a prescribed confidence level α.

Consider the general case in which constraint h constrains all m random variables in $\mathcal{S}$.

Lemma 1. *Given realisations* $\{\bar{s}_1^1, \ldots, \bar{s}_m^1\}$, $\{\bar{s}_1^2, \ldots, \bar{s}_m^2\}$, …, $\{\bar{s}_1^N, \ldots, \bar{s}_m^N\}$, *where* $\bar{s}_j^k$ *is the realised value for random variable* j *observed in the* k*-th set of realisations,* τ_i *is a deterministic test.*

Proposition 5. τ_i *is a random function of* $\{s_1, \ldots, s_m\}$ *and* N.

Proof. Immediately follows from lemma 1 and from the fact that $s_j^1, \ldots, s_j^N$ are N i.i.d. random variables. □

Proposition 6. *The maximum cardinality of an orthogonal set of random variables in* $\tau_1, \ldots, \tau_T$ *is* m.

Proof. Recall that two random variables τ_i and τ_j are orthogonal if their covariance is zero. The result follows from proposition 5. □

Proposition 7. *The maximum cardinality of an independent set of random variables in* $\tau_1, \ldots, \tau_T$ *is* m.

Proof. If two random variables are independent then they are necessarily orthogonal. A proof by contradiction can be constructed by assuming that there are $m + 1$ independent random variables. □

Proposition 8. *The probability that* $\tau_1, \ldots, \tau_m$ *are all equal to 0 is at least* $1 - m(1 - \alpha)$.

Proof. Consider the case in which events $\tau_i = 1$ and $\tau_j = 1$ are mutually exclusive for all $i, j = 1, \ldots, m$, $i \neq j$; of course it is still true that $\Pr\{\tau_i = 1\} = \Pr\{\tau_j = 1\} = 1 - \alpha$. The probability that $\tau_i = 0$ for all $i = 1, \ldots, m$ is thus $1 - m(1 - \alpha)$. Note that this is worse than the case of m independent tests, for which the probability that all tests succeed would be $\alpha^m \geq 1 - m(1 - \alpha)$. □

Proposition 9. *The probability that $\tau_1, \ldots, \tau_T$ are all equal to 0 is at least $1 - m(1 - \alpha)$.*

Proof. Follows from propositions 6 and 8. □

The property discussed in proposition 9 applies to each chance constraint $h \in C$. Let m_h be the number of random variables constrained by h, to compute an (α, ϑ)-solution set we introduce a Bonferroni's correction, which is *free of correlation and distribution assumptions*, while computing N.

Definition 5. *N is computed as the minimum value for which*

$$\max(p^{\beta}_{ub} - \beta, \beta - p^{\beta}_{lb}) \leq \vartheta,$$

where $(p^{\beta}_{lb}, p^{\beta}_{ub})$, is the Clopper-Pearson confidence interval for a confidence probability $\widehat{\alpha}$, where

$$\widehat{\alpha} = 1 - \frac{1 - \alpha}{\sum_{h \in C} m_h},$$

and round(βN) "successes" in N trials.

Definition 6. *An (α, ϑ)-solution set to a SCSP $\mathcal{P}$ is a set of policy trees. All policy trees in this set simultaneously provide, with probability at least α, a satisfaction probability greater or equal to $\beta_i - \vartheta$ for every chance constraint h_i in $\mathcal{P}$ with satisfaction threshold β_i.*

Proposition 10. *A set of policy trees that are solutions to $\widehat{\mathcal{P}}$ for a sample size N computed as discussed in definition 5 is an (α, ϑ)-solution set for $\mathcal{P}$.*

Proof. Bonferroni's correction, introduced in definition 5, ensures that for every constraint in the model the probability $\tau_1, \ldots, \tau_T$ are all equal to 0 is at least α. □

Note that, by using proposition 7, if $s_1, \ldots, s_m$ are known to be independent Bonferroni's correction can be replaced by the less conservative Šidák correction, thus leading to a smaller sample size N. Furthermore, it is worth noting that the correction in definition 5 is valid even if the feasibility of each constraint in the model is assessed against a different sample set.

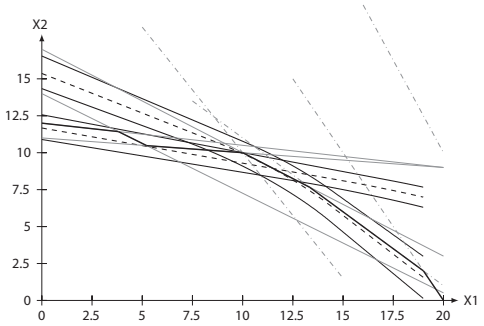

Figure 7: The piecewise boundary of an (α, ϑ)-solution set for example 4 computed for $N = 4$ samples

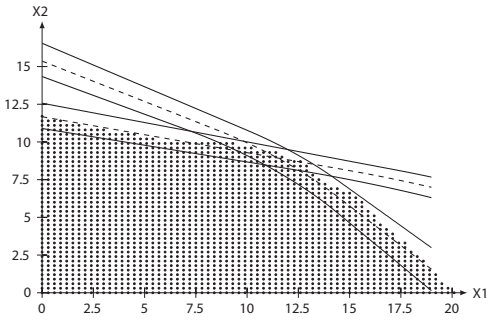

Figure 8: An (α, ϑ)-solution set for example 4 computed for $N = 348$ samples

Example 4. *Consider the following SCSP $\mathcal{P} = \langle V, S, D, P, C, \beta_c, L \rangle$, where $V = \{X_1, X_2\}$, $S = \{r_1, r_2\}$, $D(X_1) = D(X_2) = \{0, 0.01, 0.02, \ldots, 24.99, 25\}$, $D(r_1) = (0, 10)$, $P(r_1) = \text{uniform}(0, 10)$,*

$D(r_2) = (0, 30)$, $P(r_2) = \text{uniform}(0, 30)$, $D(r_3) = (0, 15)$, $P(r_3) = \text{uniform}(0, 15)$, $D(r_4) = (0, 20)$, $P(r_2) = \text{uniform}(0, 20)$, $C = \{c_1 : C_1 \geq X_1 r_1 + X_2 r_2, c_2 : C_2 \geq X_1 r_3 + X_2 r_4\}$, $\beta_{c_1} = \beta_{c_2} = 0.7$, *and* $L = [\langle V, S \rangle]$. $C_1 = 245$ *and* $C_2 = 215$ *are constants. We set* $\alpha = 0.9$ *and* $\vartheta = 0.05$. *We apply definition 5 to compute the number of samples* $N = 348$ *required to obtain an* (α, ϑ)*-solution set to* $\mathcal{P}$*; note that there are two constraints each of which constrains two random variables. We computed analytically the true boundaries of* c_1 *and* c_2 *(see [14, 15]), each of which is denoted by a dashed line in Fig. 7 and 8. We also computed confidence bands around these two dashed lines. The upper confidence band is the set of solutions that provide a satisfaction probability of exactly* $\beta_i - \vartheta$*; the lower confidence band is the set of solutions that provide a satisfaction probability of exactly* $\beta_i + \vartheta$*. In Fig. 7 we illustrate how the random boundary of an* (α, ϑ)*-solution set is generated. Each grey solid line represent the boundary of an instance of* c_1 *for a given sample. Each dash-dotted grey line is the boundary of an instance of* c_2 *for a given sample. For illustration purposes, we assume that there are only four samples considered for each constraint. The solid black piecewise boundary is the set of assignments that satisfy* c_1 *and* c_2 *in at least* $\lceil 0.7 \cdot 4 \rceil = \lceil 2.8 \rceil = 3$ *scenarios. An* (α, ϑ)*-solution set computed for* $N = 348$ *samples is shown in Fig. 8. Furthermore, we generated 1000 different instances and analytically inspected, for each of them, if the* (α, ϑ)*-solution set generated was fully contained within the upper confidence band in Fig. 8; the result of this simulation study revealed that the* (α, ϑ)*-solution set was not fully contained within the upper confidence band with probability 0.894, 0.95 confidence interval* $(0.873, 0.912)$*; this misclassification rate is in line with the prescribed* α*. Finally, it is worth noting that the random boundary of an* (α, ϑ)*-solution set is guaranteed to remain within the channel identified by the two solid confidence bands with probability* $1 - 2(1 - \alpha)$.

8. Stochastic constraint optimisation problems

The concepts introduced in sections 6 and 7 can be employed to approximate optimal solution to sampled SCOPs. In this setting, we must distinguish two possible cases: the case in which the objective function is deterministic and that in which the objective function is stochastic.

If the objective function is deterministic, it is possible to exploit the results in section 7 to obtain a confidence interval for the cost of an optimal plan.

Without loss of generality, we discuss the case in which our aim is to maximise a deterministic objective function f of the decision variables in V. Consider an SCOP $\mathcal{P} = \langle V, S, D, P, C, \beta_c, L, f \rangle$. Choose α and ϑ and construct two new SCOPs: $\mathcal{P}_1 = \langle V, S, D, P, C, \beta_c^1, L, f \rangle$, where for all $c \in C$, $\beta_c^1 = \beta_c + \vartheta$; and $\mathcal{P}_2 = \langle V, S, D, P, C, \beta_c^2, L, f \rangle$, where for all $c \in C$, $\beta_c^2 = \beta_c - \vartheta$.

Proposition 11. *an* (α, ϑ)*-solution set to* $\mathcal{P}_1$ *underestimates the true optimal profit with probability* α*; an* (α, ϑ)*-solution set to* $\mathcal{P}_2$ *overestimates the true optimal profit with probability* α.

Proof. The proof trivially follows from definition 6. □

Proposition 11 can be exploited to generate a confidence interval for the true optimal profit via a binomial reasoning. We solve M independently generated instances of $\mathcal{P}_1$ and store the optimal profit obtained for each of these instances into an array K_1 sorted in ascending order; we solve M independently generated instances of $\mathcal{P}_2$ and store the optimal profit obtained for each of these instances into an array K_2 sorted in ascending order. Let $\text{bin}^{-1}(M - 1, 1 - \alpha)$ be the inverse cumulative distribution of a binomial distribution with M trials and a success probability α; let k_1 be the $1 - (1 - \alpha)/2$-quantile of this distribution; finally, let k_2 be the $(1 - \alpha)/2$-quantile

of $\text{bin}^{-1}(M-1,\alpha)$. With confidence α element at position k_1 of K_1 is a lower bound and element at position k_2 of K_2 is an upper bound to the true optimal cost.[6]

Example 5. *We transform the SCSP in example 4 into two SCOPs $\mathcal{P}_1$ and $\mathcal{P}_2$ that maximise the objective function $f(X_1, X_2) = X_1 + 2X_2$. In other words, we assume the the profit per unit of X_1 is 1 and the profit per unit of X_2 is 2. By choosing $M = 20$ we obtain the α confidence interval $(282, 304)$ for the true optimal profit 293; if we reduce ϑ to 0.01 the interval shrinks considerably to $(290, 294)$.*

If the objective function is stochastic there is not a unique way to proceed. For instance, based on the available samples one may derive standard confidence intervals for the expected value of a stochastic expression based on the Student's T distribution and then compare solutions or partial assignments by comparing upper or lower limits of these intervals. An example of a filtering algorithm that may be employed in such context is discussed in Appendix A. We will make use of this propagator to solve the models discussed in section 10. However, this is not the only possible way to proceed. Other approaches for handling stochastic objective functions will be briefly discussed in section 12.

9. Connections with statistics

To better understand the concepts just introduced, it is worth discussing the connection between the approach introduced and hypothesis testing in statistical analysis. Let us assume that our "null hypothesis" (H_0), in statistical sense, is that an assignment is feasible. According to classical hypothesis testing we may have four cases, as illustrated in Table 1. We may have a feasible assignment at hand (H_0 true) and we may incorrectly filter it (Type I error); or we may be operating on an infeasible assignment (H_0 false) and we may fail to reject it (Type II error).

	H_0 is true	H_0 is false
Reject H_0	Type I error (false positive)	Correct outcome (true positive)
Fail to reject H_0	Correct outcome (true negative)	Type II error (false negative)

Table 1: Type I and Type II errors in statistics

In clinical trials or quality control, it is key to control the rate of Type I errors. It is undesirable to put under treatment a healthy a patient or to discard an expensive machine that is working fine. However, there are cases in which controlling Type II errors is essential. For example, aerospace engineers would prefer to throw an electronic circuit that is working fine than to use one on a spacecraft that is actually broken, in such a situation a Type I error raises the budget, but a Type II error would risk the entire mission. In general, minimising Type I and Type II errors is not a simple issue; for any given sample size the effort to reduce one type of error generally results in increasing the other type of error. The only way to minimise both types of error, is to increase the sample size. If one tries to reduce the rate of occurrence for Type I errors, the direct consequence is typically an increase in the observed rate for Type II errors and vice-versa. So in practice, one tries to control either Type I or Type II errors and, if the rate of the type that is not controlled is too high, then increases the sample size.

[6] Elements of K_i are indexed as follows: $0, 1, \ldots, |K_i| - 1$

In our specific case it is clearly essential to control the rate of Type II errors, which are more delicate than Type I errors. Making a Type II error means retaining an infeasible assignment, which is what we want to avoid as much as possible. Making a Type I error means discarding a feasible solution, which may impact optimality for an optimisation problem, or may lead to an empty solution space. Since our approach is essentially a heuristic, it is clear that both these issues — a poor solution quality or an empty solution space — are acceptable and should be dealt with by increasing the number of samples.

10. Computational experience

The aim of this section is to provide numerical insights on the theoretical framework introduced and particularly on the concepts of (α,ϑ) solution set and on its applications to find approximate solution to SCSPs and SCOPs. In our numerical study we will consider three well-known problems: the static stochastic knapsack (Section 10.1), the stochastic multiprocessor scheduling problem with release time and deadlines (Section 10.2), and the static stochastic lot-sizing (Section 10.3). The first and the third problems are single-stage, while the second is two-stage. In Section 10.4 we will generate (α,ϑ) solution sets for the first two problems and show numerically that, with probability greater or equal to α, the approach we discussed generates solution sets that satisfy chance constraints in the model with a margin of error ϑ. In Section 10.5 we will numerically illustrate that the upper and lower profit/cost bounds obtained with the approach outlined in Section 8 comply with the prescribed confidence level α and we will also show the behaviour of the optimality gap as a function of the chosen error threshold ϑ and number of replications M.

10.1. Static stochastic knapsack

The knapsack problem [16] is a well-known combinatorial optimisation problem. The decision maker is given a set of objects each of which is associated with a weight and a profit. The aim is then to select a subset of these objects that fit into a given capacity and bring the maximum profit. As discussed in [17] there are several possible stochastic variants of the knapsack problem. Stochastic versions of the knapsack problem into static or dynamic. In the static stochastic knapsack problem, object weights and/or profits are random and the decision maker must choose, before observing any of their weights/profits, a subset of these objects that maximises a given objective, e.g. the expected profit, while meeting a restriction, e.g. a chance constraint, on the given capacity. Conversely, in the dynamic stochastic knapsack, the decision maker selects an object and immediately observes its weight and/or profit, based on this information she can then decide if selecting or not other objects.

In our computational study we will consider the SCSP presented in Fig. 9, i.e. a static stochastic multiple knapsack (SSMKP). In this problem we have a set of N types of objects; there are D objects of type i available. Each object of type i is associated with G random "coefficients" s_i^k; these coefficients follow a Poisson distribution with mean λ_i^k and appear in the context of G chance constraints. The first L of these chance constraints are of type (1), i.e. they can be seen as "capacity restrictions" with respect to a target capacity C_k, and they should be satisfied with probability β. The remaining $G - L$ restrictions are of type (2), i.e. they can be seen as "minimum production requirements" with respect to a target level C_k, and again they should be satisfied with probability β. Our aim is to determine the feasible region of the problem, i.e. the set of assignments that satisfy constraints (1) and (2).

Constraints:
(1) $\Pr\{s_1^k x_1 + \ldots + s_N^k x_N \leq C^k\} \geq \beta \quad k = 1, \ldots, L$
(2) $\Pr\{s_1^k x_1 + \ldots + s_N^k x_N \geq C^k\} \geq \beta \quad k = L+1, \ldots, G$

Decision variables:
$x_i \in \{0, \ldots, D\} \quad i = 1, \ldots, N$

Random variables:
$s_i^k \leftarrow \text{Poisson}(\lambda_i^k) \quad i = 1, \ldots, N; k = 1, \ldots, G$

Stage structure:
$V_1 = \{x_1, \ldots, x_N\}$
$S_1 = \{s_1^1, \ldots, s_n^k, \ldots, s_N^G\}$
$L = [\langle V_1, S_1 \rangle]$

Figure 9: The static stochastic multiple knapsack as an SCSP

Objective:
(1) $\max \mathrm{E}[p_1 x_1 + \ldots + p_N x_N]$

Constraints:
(2) $\Pr\{s_1^k x_1 + \ldots + s_N^k x_N \leq C^k\} \geq \beta \quad k = 1, \ldots, L$

Decision variables:
$x_i \in \{0, \ldots, D\} \quad i = 1, \ldots, N$

Random variables:
$s_i^k \leftarrow \text{Poisson}(\lambda_i^k) \quad i = 1, \ldots, N; k = 1, \ldots, L$
$p_i \leftarrow \text{Poisson}(\pi_i) \quad i = 1, \ldots, N$

Stage structure:
$V_1 = \{x_1, \ldots, x_N\}$
$S_1 = \{s_1^1, \ldots, s_n^k, \ldots, s_N^L\}$
$L = [\langle V_1, S_1 \rangle]$

Figure 10: The static stochastic multiple knapsack as an SCOP

We will also consider an optimisation version of the problem (Fig. 10) in which our aim is to determine what subset of objects maximises the expected total profit. Therefore we will consider a random profit p_i, which follows a Poisson distribution with mean π_i, for each of the N objects.

10.2. Stochastic multiprocessor scheduling problem with release time and deadlines

We consider a multiprocessor scheduling problem (MPSP, see [18], p. 238). The problem consists in finding a feasible schedule to process a set of K orders (or jobs) using m processors, where $m \leq P$. Processing an order k can only begin after the release date r_k and must be completed at the latest by the due date d_k. Order k requires a certain capacity c_k — expressed in terms of the number of processors — to be processed. The processing time of order k is t_k. The problem just described is well known in scheduling and it is fully deterministic and can easily and compactly be modelled using the `cumulative` constraint [19]. Let the height of a task k be c_k. This constraint considers a set of tasks and enforces that at each point in time the cumulated height of the set of tasks that overlap that point does not exceed a given limit m. A task k overlaps a point i if and only if its origin s_k is less than or equal to i, and its end e_k is strictly greater than i. This constraint also imposes, for each task k, the constraint $s_k + t_k = e_k$.

However, in reality, some parameters of this problem are uncertain in nature. Jobs may take longer than expected, some processors may break down and become unavailable, the release and due dates may be delayed, etc. In order to better model this problem a number of stochastic generalizations may be considered such as uncertain release date r_k; uncertain due date d_k; uncertain processing capacity c_k; uncertain processing time t_k; and uncertain number m of available processors; and every possible combination stemming from these cases.

We will consider the following stochastic constraint programming formulation of the stochastic multiprocessor scheduling problem (SMPSP), in which only processing time t_k for order k is uncertain; this is shown in Fig. 11. In this model, decision variables s_k and e_k denote the start time and the completion time of each job k, respectively. The processing time t_k of each job k is modeled as a Poisson distributed random variable with mean λ_k. In contrast to the problem presented in Section 10.1, this model is a two-stage SCSP. In the first stage, we decide on the start time of each job then we observe the realisation of the processing time. In the second stage the completion times are decided. Under this stage structure, constraint (1) enforces that the *probability* of not exceeding the given deadline for each job and the number of available processors m stays above the specified threshold β. More specifically, this constraint is a global chance constraint embedding a well-known global constraint: the `cumulative` constraint [19]. This constraint can be filtered using the general purpose method discussed in [4] .

In our computational study we will also consider an optimisation version of the above problem in which we aim to minimise the latest start time.

Constraints:
(1) $\Pr\{\text{cumulative}(s, e, t, c, m)\} \geq \beta$

Decision variables:
$s_k \in \{r_k, \ldots, d_k\}$, $\forall k \in 1, \ldots, K$
$e_k \in \{r_k, \ldots, d_k\}$, $\forall k \in 1, \ldots, K$

Stochastic variables:
$t_k \rightarrow \text{Poisson}(\lambda_k)$ $\forall k \in 1, \ldots, K$

Stage structure:
$V_1 = \{s_1, s_2, \ldots, s_K\}$ $S_1 = \{t_1, t_2 \ldots, t_K\}$
$V_2 = \{e_1, e_2, \ldots, e_K\}$ $S_2 = \{\}$
$L = [\langle V_1, S_1\rangle, \langle V_2, S_2\rangle]$

Figure 11: An SCSP for the stochastic multiprocessor scheduling problem with release time and deadlines

Objective:
(1) $\min \text{E}[\sum_{t=1}^{N}(a\delta_t + h\sum_{j=1}^{t}(Q_t - d_t))]$

Constraints:
(2) $\Pr\{\sum_{j=1}^{t}(Q_t - d_t) \geq 0\} \geq \beta$ $t = 1, \ldots, T$
(3) $\delta_t = 0 \implies Q_t = 0$ $t = 1, \ldots, T$

Decision variables:
$\delta_t \in \{0, 1\}$ $t = 1, \ldots, T$
$Q_t \in \{0, \ldots, C\}$ $t = 1, \ldots, T$

Random variables:
$d_t \leftarrow \text{Poisson}(\lambda_t)$ $t = 1, \ldots, T$

Stage structure:
$V_1 = \{Q_1, \ldots, Q_T, \delta_1, \ldots, \delta_T\}$
$S_1 = \{d_1, \ldots, d_T\}$
$L = [\langle V_1, S_1\rangle]$

Figure 12: The stochastic lot sizing problem in [20] as an SCOP (static uncertainty strategy)

10.3. Static stochastic lot-sizing

The last problem we will consider in our computational study is the stochastic lot sizing problem introduced in [20]. An SCOP for this problem is shown in in Fig. 12. The decision maker faces a finite horizon of T periods and a random demand d_t in each period; which without loss of generality we will consider Poisson distributed with mean λ_t. There is a fixed cost a for placing an order of size $0 < Q_t \leq C$ in period t. An order placed in period t is delivered immediately at the beginning of the period, before demand occurs. Binary decision variable δ_t is set to zero if no order is placed (3). There is a holding cost h charged on items that are carried over from one period to the next. Finally, the decision maker must comply with a service level restriction (2) stating that the net inventory at the end of each period should be nonnegative with probability at least β. The aim is to meet these service level restrictions while minimising the expected total cost (1).

The authors in [20] describe a range of control policies that can be used to control such a system. In our study, we will adopt the static uncertainty policy, which fixed all Q_t's and δ_t's at the beginning of the planning horizon, before demand is observed. Note that other strategies discussed in [20], i.e. dynamic uncertainty and static-dynamic uncertainty, can be easily captured by modifying the stage structure of the SCOP. In what follows, we shall refer to this problem as the static stochastic lot sizing problem (SSLSP).

10.4. Feasibility

In Section 7 we introduced the notion of (α,ϑ) solution set. We will now present a computational analysis for the SCSPs presented in Sections 10.1 and 10.2 demonstrating that, with probability α, the approach we discussed generates solution sets that satisfy chance constraints in the model with a margin of error ϑ.

We considered thirty randomly generated small instances of the single stage problem in Fig. 9 in which $N = 2$, $L = 2$, $G = 3$, $D = 250$ and $\beta = 0.7$. Means λ_i^k of random variables in the model were integer numbers uniformly distributed between 10 and 20 for constraints (1)

and between 20 and 30 for constraints (2). Right hand side constants C^k were integer numbers uniformly distributed between 1500 and 2000 for constraints (1) and between 2500 and 3000 for constraints (2). We fixed $\alpha = 0.9$ and $\vartheta = 0.2$, according to Definition 5 this choice led to a sample size of 31.

We also considered thirty randomly generated small instances of the two stage problem in Fig. 11 in which $K = 2$ and $\beta = 0.6$. Domains of decision variable r_k and d_k, which represent job release times and deadlines respectively, were generated as $\{0, \ldots, R_k\}$ and $\{0, \ldots, D_k\}$, where R_k and D_k were both set to 4. Capacity requirements c_k where generated as integer numbers uniformly distributed between 1 and 2. Finally, expected task durations λ_k were generated as uniformly distributed numbers between 1 and 3; the maximum number of processors P was set to 3. We fixed $\alpha = 0.9$ and $\vartheta = 0.35$, this choice led to a sample size of 6.

Instances were purportedly small since in our analysis we generated the complete set of feasible assignments of the respective sampled SCSP, i.e. an (α, ϑ) solution set, which for the two-stage problem in Fig. 11 was generally extremely large, in the order of tens of thousands solutions. Feasibility of each of these assignment with respect to the original SCSP was then assessed via Monte Carlo simulation; the number of replications was set in such a way as to guarantee a margin of error of $\vartheta/10$ with a confidence level of 0.9 — so that the Monte Carlo simulation error is an order of magnitude smaller than the approximation error associated with the (α,ϑ) solution set obtained.

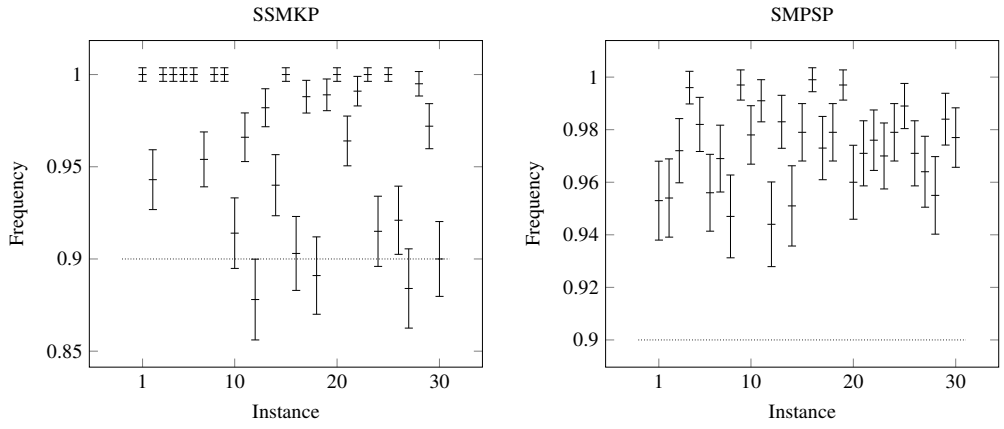

Figure 13: Frequency of event "all feasible assignments of the sampled SCSP are feasible with respect to the original SCSP within the given tolerance threshold ϑ" over 1000 sampled SCSPs; together with the frequency, we report the associated confidence interval (confidence level of 0.95)

To numerically investigate if those computed are effectively (α, ϑ) solution sets, for each of the above sixty instances, we repeatedly solved 1000 sampled SCSPs and computed the frequency of event e: "**all feasible assignments** of the sampled SCSP are feasible with respect to the original SCSP within the given tolerance threshold ϑ." In Fig. 13, for both problems and for each instance, we report the frequency of event e and the associated confidence intervals (confidence level of 0.95). These frequencies, are in line with the claim that those computed are (α, ϑ) solution sets, for the given $\alpha = 0.9$. Note that our aim is to control Type-II errors (an infeasible

assignment regarded as feasible), and not Type-I errors (a discarded and yet feasible assignment); for this reason if the sampled SCSP admitted no solution, this was regarded as a degenerate case in which all feasible assignments (i.e. none) of the sampled SCSP were feasible with respect to the original SCSP within the given tolerance threshold ϑ. Finally, it is worth observing that some of the frequencies observed in Fig. 13 are strictly greater than the prescribed value α. This is due to the fact that only assignments providing a satisfaction probability of exactly $\beta - \vartheta$ are correctly classified as infeasible with probability α. However, given the discrete nature of the assignment space, it is likely that instances may not feature any of such assignments. Assignments providing a satisfaction probability strictly less than $\beta - \vartheta$ are correctly classified as infeasible with probability strictly greater than α. In addition to this, when a model features multiple chance constraints, Bonferroni's correction, which is free of correlation and distribution assumptions, might generate a conservative — i.e. strictly larger than needed — sample size.

10.5. Optimality

We considered fifty randomly generated small instances of the problem in Fig. 10 in which $N = 10$, $L = 2$, $D = 1$ and $\beta = 0.9$. Means λ_i^k of random variables in the model were integer numbers uniformly distributed between 10 and 20 for constraints (1). Right hand side constants C^k were integer numbers uniformly distributed between 100 and 200 for constraints (1). Means π_i were all set to 10.

We also considered fifty randomly generated small instances of the the problem in Fig. 12 in which $T = 5$, $h = 1$, $a = 10$, $C = 100$, and $\beta = 0.9$. Means λ_i^t of Poisson demand in each period $t = 1, \ldots, T$ were integer numbers uniformly distributed between 5 and 10.

We fixed $\alpha = 0.9$, $\vartheta = 0.05$ and $M = 10$; recall that M is the number of replications used for computing profit/cost upper and lower bounds as illustrated in Section 8.

Due to the small size of the SSMKP instances, we managed to obtain optimal solutions by exhaustive enumeration, i.e. we generated all possible assignment and then checked feasibility and expected total profit of each of them via Monte Carlo simulation. The number of Monte Carlo runs was set to guarantee a margin of error of $\vartheta/10$ with a confidence level of 0.9; in such a way as to ensure an approximation error negligible with respect to the chosen ϑ. SSLSP instances can be solved to optimality by using a deterministic equivalent mixed integer linear programming model [21]. In our analysis, we can therefore compare results obtained with our approach against the true optimal solutions.

In Fig. 14, for each instance, we plotted upper and lower bound obtained for its optimal profit (SSMKP) or cost (SSLSP). For clarity, the interval has been normalised by using the profit/cost of the true optimal solution as a normalisation factor, so that value one in the graph denotes the true optimal profit/cost. The confidence level achieved by using our approach is generally higher than the prescribed α. In fact, despite α being set to 0.9, over the hundred instances analysed, the cost confidence interval did not cover the true optimal cost only in one case (SSMKP, instance 18). This is due to the conservative nature of our approach, as already discussed in Section 10.4.

Finally, we included in the analysis randomly generated instances of the SMPSP formulated as an SCOP in which the objective is to minimise the latest start time. In these instances $K = 5$ and $\beta = 0.6$. Domains of decision variable r_k and d_k, which represent job release times and deadlines respectively, were generated as $\{0, \ldots, R_k\}$ and $\{0, \ldots, D_k\}$, where R_k and D_k were both set to 20. Capacity requirements c_k where generated as integer numbers uniformly distributed between 1 and 3. Expected task durations λ_k were generated as uniformly distributed numbers between 1 and 5; the maximum number of processors P was set to 5.

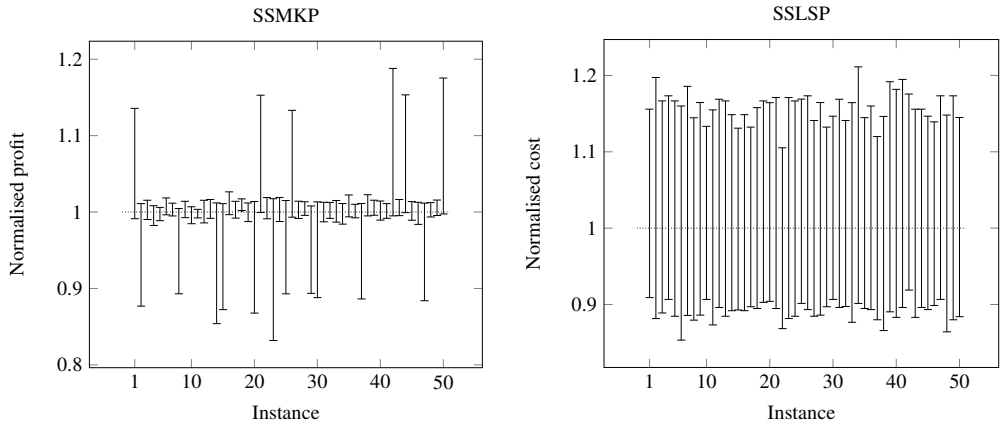

Figure 14: Normalised profit/cost upper and lower bounds for fifty SSMPSP and SSLSP instances; a value of 1 denotes the true optimal profit/cost.

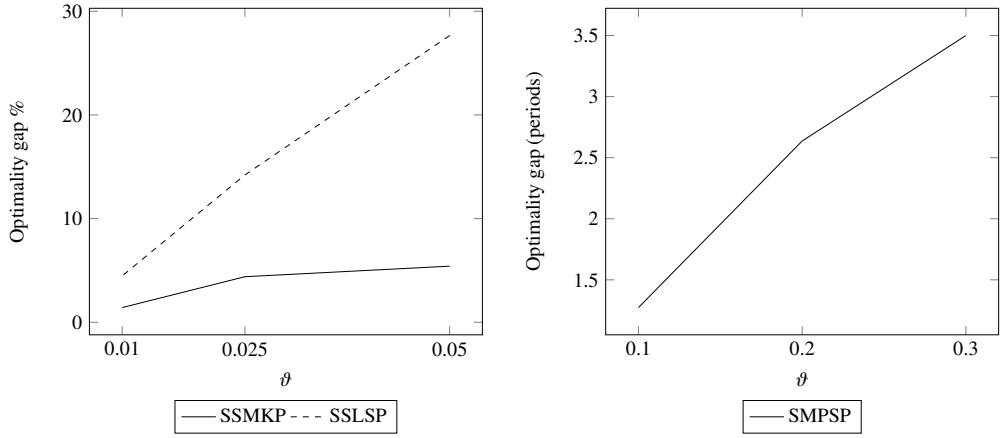

Figure 15: Average optimality gap for different values of ϑ

In Fig. 15 we analysed the behaviour of the optimality gap when $\alpha = 0.9$, $M = 10$ and ϑ varies. For each value of ϑ considered, we solved 10 different instances of the SSMKP, SSLSP and SMPSP, and we computed the average optimality gap over this pool of instances. The average optimality gap for the SSMKP and the SSLSP is reported in percentage of the true optimal solution. For the case of the SMPSP unfortunately we were not able to compute the true optimal plan, therefore we reported the optimality gap in absolute terms; since we are minimising the latest start time, we expressed the optimality gap in expected number of periods. Note that since α and ϑ are linked to the number of samples generated by the relation in Lemma 5, similar plots may be obtained by varying α and keeping ϑ fixed. In Fig. 15 we carried out a similar analysis

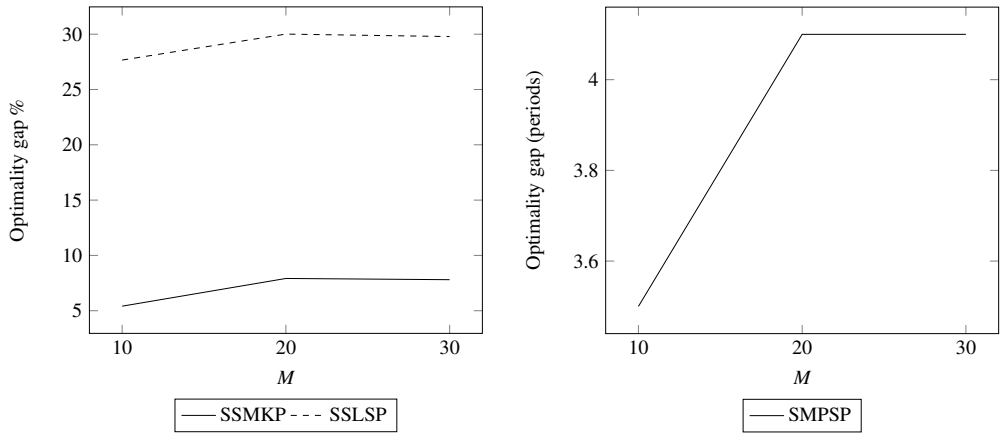

Figure 16: Average optimality gap for different values of M

by keeping ϑ fixed to 0.05 (SSMKP, SSLSP) and to 0.3 (SSMKP) and by varying M. It is worth observing that increasing M does not seem to positively affect the optimality gap and that a small value of M provides the best results in our study.

11. Related works

A detailed discussion on hybrid CP/AI/OR approaches for decision making under uncertainty can be found e.g. in [22, 23]. We direct the reader to these key references for further details on existing works in this research area.

Confidence-based optimisation was originally introduced in [24], where the authors discussed an application of this methodology in the context of a well-known stochastic inventory control problem. Our work extends the discussion there presented to generic SCSPs and SCOPs by introducing a more general notion of confidence-based reasoning based on two novel concepts: (α, ϑ) solutions and (α, ϑ) solution sets. In the context of stochastic modeling and optimisation, as discussed, these tools can be employed to find solutions that feature given statistical properties.

In what follows we will summarise existing works that are closely related to the techniques proposed in this work.

11.1. Related works in stochastic programming

In operations research, and particularly in stochastic programming, the state-of-the-art technique that applies sampling in combinatorial optimization is the Sample Average Approximation (SAA) approach [25]. In this approach a given number of samples is drawn from the random variable distributions, and the combinatorial problem of interest is repeatedly solved by considering different samples as input in each run. The real expected cost/profit of a solution produced for a given sample is then computed by simulating a sufficient number of samples. Among all the solutions computed, the one that provides the minimum expected cost (or the maximum expected profit) is retained. Two criteria are given by the authors: one for deciding when a given sample

size is no more likely to produce better solutions, and one to decide if increasing the sample size may lead to better solutions. Extensions such as those discussed in [26] have been proposed for dealing with chance-constrained stochastic programs. Nevertheless, SAA typically operates by iteratively solving a number of sampled average approximation problems usually formulated as integer linear programs. To the best of our knowledge, no concept that resembles that of (α,ϑ)-solution (set) can be found in the SAA seminal work and in any of the proposed extensions in the literature. In fact, none of these extensions is based on confidence interval analysis. Typically, the analysis conducted show asymptotical convergence properties of the estimators employed based on inequalities such as Chernoff's [27] or Hoeffding's [28].

11.2. Related works in constraint programming

Efforts that try to extend classical CSP framework to incorporate uncertainty have been influenced by works that originated in different fields, namely *chance-constrained programming* [29] and *stochastic programming* [30]. To the best of our knowledge the first work that tries to create a bridge between Stochastic Programming and Constraint Programming is by Benoist et al. [31]. Search and consistency strategies, namely a backtracking algorithm, a forward checking procedure [5] and an arc-consistency [32] algorithm have been proposed for SCSPs. A scenario-based approach for building up constraint programming models of SCSPs was proposed by Tarim et al. [6]. In the same work a fully featured language — Stochastic OPL — for modeling SCSPs was also proposed. Global chance constraints were introduced first in [8], they bring together the reasoning power of global constraints from CP and the expressive power of chance constraints from SP. A general purpose approach for filtering global chance constraints is proposed in [3]. This approach is able to reuse existing propagators available for the respective deterministic global constraint which corresponds to a given global chance constraint when all the random variables are replaced by constant parameters. In [33] the authors discuss some possible strategies to perform cost-based filtering for certain classes of Stochastic COPs. These strategies exploit well-known inequalities borrowed from SP and used to compute valid bounds for any given Stochastic COP that respects some mild assumptions.

In [6] the authors employed sampling in order to reduce the number of scenarios considered for a given stochastic constraint program and produce a solution in reasonable time. Nevertheless, this approach does not provide any guarantee for the degree of optimality as well as for the feasibility of the solution produced. Forward sampling [31, 34] and sample aggregation [35] are two other techniques that have been employed to solve SCSPs. Nevertheless, none of these approaches introduce a concept that resembles that of (α,ϑ)-solution. Probably, the work discussed in [36] represents the closest attempt to provide some sort of guarantees for a stochastic constraint satisfaction problem. Nevertheless, this work is focused on a specific problem — a two-stage stochastic matching problem — and it does not propose a generic approach for solving SCSPs.

12. Extensions to the framework

In what follows we briefly discuss a number of suggestions for future work.

Stochastic objective function. In Section 8 we introduced a possible strategy for dealing with SCOPs in which the objective function involves random profits or costs. As pointed out, in this setting there is not a unique way to proceed. Our choice in this work was to exploit the filtering

algorithm discussed in Appendix A. This algorithm is designed to handle the situation in which the objective is to minimise/maximise the expected value of some expression involving decision and random variables. Of course, different algorithms must be designed if the objective involves a different operator, e.g. variance. Our algorithm distinguishes the case in which we are trying to determine an upper or a lower bound for the expected cost of an optimal solution. It then exploit the sampling distribution (i.e. Student's T distribution) of the expected total profit/cost and filters values based on upper/lower confidence limits obtained via this distribution. For instance, if our aim is to determine an upper bound for the optimal profit (problem type $\mathcal{P}_2$), our algorithm will simply compare the upper confidence limits of the expected profit of two assignments and retain the assignment with the highest upper confidence limit. A different strategy may instead compare not only the upper confidence limits, but the whole intervals. An assignment would then provide a lower/higher profit than another if and only if their profit confidence intervals do not overlap. However, due to the complexity of the filtering logic that would be required in this case, we prefer to leave this discussion as future work.

Rolling horizon. A promising direction is that of employing our approach within a "rolling horizon" framework [10]. Such a strategy has clear connections to online stochastic optimization [31] and it may enhance the results in [35, 37, 38, 39] by ensuring a better control of the solution quality obtained at each step of the online process.

Sampling strategies. Another open issue is related to the fact that simple random sampling [40] is a relatively naive strategy for selecting samples. The use of more refined sampling strategies — for instance a stratified sampling technique such as Latin Hypercube Sampling [41] — may of course reduce the number of samples required to produce an (α,ϑ)-solution. Nevertheless, further research is required in order to clarify how stratified sampling can be effectively employed in this context.

Confidence intervals. The Clopper-Pearson interval is an exact interval since it is based directly on the binomial distribution rather than any approximation to the binomial distribution. This interval, however, can be conservative because of the discrete nature of the binomial distribution, as pointed out by Neyman [42]. For example, the true coverage rate of a 95% Clopper-Pearson interval may be well above 95%, depending on n and q. Thus the interval may be wider than it needs to be to achieve 95% confidence. In contrast, it is worth noting that other confidence bounds may be narrower than their nominal confidence width, i.e., the Normal Approximation Interval also known as Wald confidence interval, Wilson Interval, Agresti-Coull Interval, etc, with a nominal coverage of 95% may in fact cover less than 95% [12]. Future research may investigate the application of approximate intervals in the context of sample-based constraint solving. The performance of each of these approximate intervals have been thoroughly analyzed in the existing body of literature. Approximate intervals may lead to smaller sample sets and therefore to more compact sampled SCSPs.

13. Conclusions

We proposed a framework for exploiting sampling in order to solve SCSPs that include random variables over a continuous or very large discrete support. Our framework is based on a number of novel concepts: sampled SCSPs, (α, ϑ)-solutions and (α, ϑ)-solution sets. We employed statistical estimation to determine if a given assignment is consistent with respect to a

given set of chance constraints. As in statistical estimation, the quality of our estimate is determined via confidence interval analysis. In contrast to existing approaches based on sampling, we provide likelihood guarantees for the quality of the solutions found. In fact, we explicitly state a confidence probability α that bounds the probability of exceeding a given error tolerance threshold ϑ in our estimation. By properly choosing the estimation error ϑ and the confidence probability α it is possible to generate compact sampled SCSPs that can be effectively solved by existing solution methods. We also extended the reasoning to SCOPs and demonstrated how to produce statistical upper and lower bounds for the value of the optimal solution. We demonstrated our approach on a number of SCSPs and SCOPs: the static stochastic knapsack problem, a stochastic multiprocessor scheduling problem, and a stochastic lot sizing problem. Our computational study demonstrates the effectiveness of our approach.

Appendix A. Filtering strategy for constraint expressions involving expected values

We discuss a filtering strategy for handling constraint expressions involving expected values in sampled SCSPs. This filtering strategy can be employed, in concert with the approach discussed in Section 8, to deal with the case in which the objective function is stochastic. Consider a constraint $x = \mathrm{E}[\langle \mathrm{exp} \rangle]$, where E[] denotes the expectation operator and x is a real valued decision variable, whose domain is stored as an interval with real valued upper and lower bounds. Techniques for handling propagation and search involving real valued decision variables are discussed in [43]. A filtering algorithm that enforces bounds consistency on this constraint is shown in Fig. 1. It should be noted that the approach discussed in Section 8 distinguishes two cases: the

Algorithm 1: Filtering Expected Values in sampled SCSPs

input : type; $\langle \mathrm{exp} \rangle$; $\mathcal{T}$; x; α.
output: Bound consistent x.

begin
  $U \leftarrow \{\}$; $L \leftarrow \{\}$;
  for *each* $p \in \Psi$ **do**
    $U \leftarrow U \cup \mathrm{Sup}(\langle \mathrm{exp} \rangle_{\downarrow p})$;
    $L \leftarrow L \cup \mathrm{Inf}(\langle \mathrm{exp} \rangle_{\downarrow p})$;
  $t \leftarrow \mathrm{StudentT}(|\Psi| - 1)$;
  if *type*=$\mathcal{P}_1$ **then**
    $\mathrm{Sup}(x) \leftarrow \mathrm{mean}(U) - \mathrm{CDF}_t^{-1}(1 - (1 - \alpha)/2) \cdot \mathrm{std}(U)/\sqrt{|\Psi|}$;
    $\mathrm{Inf}(x) \leftarrow \mathrm{mean}(L) - \mathrm{CDF}_t^{-1}(1 - (1 - \alpha)/2) \cdot \mathrm{std}(L)/\sqrt{|\Psi|}$;
  else if *type*=$\mathcal{P}_2$ **then**
    $\mathrm{Sup}(x) \leftarrow \mathrm{mean}(U) + \mathrm{CDF}_t^{-1}(1 - (1 - \alpha)/2) \cdot \mathrm{std}(U)/\sqrt{|\Psi|}$;
    $\mathrm{Inf}(x) \leftarrow \mathrm{mean}(L) + \mathrm{CDF}_t^{-1}(1 - (1 - \alpha)/2) \cdot \mathrm{std}(L)/\sqrt{|\Psi|}$;

one in which our aim is to underestimates the true optimal profit (SCOP $\mathcal{P}_1$) and that in which our aim is to overestimates the true optimal profit (SCOP $\mathcal{P}_2$). The type of problem ($\mathcal{P}_1$ or $\mathcal{P}_2$) which the propagator belongs to must be specified as an input parameter "type" that influences propagation. The algorithm constructs two arrays: U and L. U lists, for each scenario, an upper

bound for the expected value of $\langle \exp \rangle$, L lists, for each scenario, a lower bound for the expected value of $\langle \exp \rangle$. Then it exploits the Student's T distribution with $|\Psi| - 1$ degrees of freedom (StudentT($|\Psi| - 1$)) to determine upper and lower confidence limits for the expected value of $\langle \exp \rangle$ at the prescribed confidence level α. Note that $\mathrm{CDF}_t^{-1}(\alpha)$ denotes the inverse cumulative distribution function of t and std(X) is the standard deviation of the elements in X. The algorithm operates by exploiting the structure Ψ of the policy tree; therefore it takes implicitly into account the stage structure of the problem while computing the expected value of a given expression; therefore it will correctly evaluate expected values both in a single or multi-stage case.

References

[1] R. Rossi, B. Hnich, S. A. Tarim, S. Prestwich, Finding (α,ϑ)-solutions via sampled SCSP, in: T. Walsh (Ed.), Proceedings of the 22nd International Joint Conference on Artificial Intelligence, IJCAI 2011, Barcelona, Spain, July 16-22, AAAI Press, 2011, pp. 2172–2177.

[2] F. Rossi, P. van Beek, T. Walsh, Handbook of Constraint Programming (Foundations of Artificial Intelligence), Elsevier Science Inc., New York, NY, USA, 2006.

[3] B. Hnich, R. Rossi, S. A. Tarim, S. D. Prestwich, Synthesizing filtering algorithms for global chance-constraints, in: I. P. Gent (Ed.), Principles and Practice of Constraint Programming - CP 2009, 15th International Conference, CP 2009, Lisbon, Portugal, September 20-24, 2009, Proceedings, volume 5732 of *Lecture Notes in Computer Science*, Springer, 2009, pp. 439–453.

[4] B. Hnich, R. Rossi, S. A. Tarim, S. Prestwich, Filtering algorithms for global chance constraints, Artificial Intelligence 189 (2012) 69–94.

[5] T. Walsh, Stochastic constraint programming, in: F. van Harmelen (Ed.), European Conference on Artificial Intelligence, ECAI'2002, Proceedings, IOS Press, 2002, pp. 111–115.

[6] S. A. Tarim, S. Manandhar, T. Walsh, Stochastic constraint programming: A scenario-based approach, Constraints 11 (2006) 53–80.

[7] H. Jeffreys, Theory of Probability, Clarendon Press, Oxford, UK, 1961.

[8] R. Rossi, S. A. Tarim, B. Hnich, S. D. Prestwich, A global chance-constraint for stochastic inventory systems under service level constraints, Constraints 13 (2008) 490–517.

[9] S. D. Prestwich, S. A. Tarim, R. Rossi, B. Hnich, Evolving parameterised policies for stochastic constraint programming, in: I. P. Gent (Ed.), Principles and Practice of Constraint Programming - CP 2009, 15th International Conference, CP 2009, Lisbon, Portugal, September 20-24, 2009, Proceedings, volume 5732 of *Lecture Notes in Computer Science*, Springer, 2009, pp. 684–691.

[10] S. Sethi, G. Sorger, A theory of rolling horizon decision making, Ann. Oper. Res. 29 (1991) 387–416.

[11] C. J. Clopper, E. S. Pearson, The use of confidence or fiducial limits illustrated in the case of the binomial, Biometrika 26 (1934) 404–413.

[12] A. Agresti, B. A. Coull, Approximate is better than "exact" for interval estimation of binomial proportions, The American Statistician 52 (1998) 119–126.

[13] M. Evans, N. Hastings, B. Peacock, Statistical Distributions, Wiley, New York, 2000.

[14] S. Sadooghi-Alvandi, A. Nematollahi, R. Habibi, On the distribution of the sum of independent uniform random variables, Statistical Papers 50 (2009) 171–175.

[15] F. Killmann, E. von Collani, A note on the convolution of the uniform and related distributions and their use in quality control, Economic Quality Control 16 (2001) 17–41.

[16] S. Martello, P. Toth, Knapsack Problems: Algorithms and Computer Implementations, John Wiley & Sons, Inc., New York, NY, USA, 1990.

[17] A. J. Kleywegt, J. D. Papastavrou, The dynamic and stochastic knapsack problem, Operations Research 46 (1998) pp. 17–35.

[18] M. R. Garey, D. S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, W. H. Freeman, 1979.

[19] A. Aggoun, N. Beldiceanu, Extending chip in order to solve complex scheduling and placement problems, Math. Comput. Model. 17 (1993) 57–73.

[20] J. H. Bookbinder, J. Y. Tan, Strategies for the probabilistic lot-sizing problem with service-level constraints, Management Science 34 (1988) 1096–1108.

[21] V. Vargas, An optimal solution for the stochastic version of the wagner-whitin dynamic lot-size model, European Journal of Operational Research 198 (2009) 447–451.

[22] K. N. Brown, I. Miguel, Uncertainty and change, in: F. Rossi, P. van Beek, T. Walsh (Eds.), Handbook of Constraint Programming, Elsevier, 2006.

[23] B. Hnich, R. Rossi, S. A. Tarim, S. Prestwich, A survey on CP-AI-OR hybrids for decision making under uncertainty, in: P. van Hentenryck, M. Milano (Eds.), Hybrid Optimization, volume 45 of *Springer Optimization and Its Applications*, Springer New York, New York, NY, 2011, pp. 227–270. doi:10.1007/978-1-4419-1644-0_7.

[24] R. Rossi, S. Prestwich, S. A. Tarim, B. Hnich, Confidence-based optimisation for the newsvendor problem under binomial, poisson and exponential demand, European Journal of Operational Research (2014).

[25] A. J. Kleywegt, A. Shapiro, T. Homem-De-Mello, The sample average approximation method for stochastic discrete optimization, SIAM Journal of Optimization 12 (2001) 479–502.

[26] S. Ahmed, A. Shapiro, Solving chance-constrained stochastic programs via sampling and integer programming, in: Z.-L. Chen, S. Raghavan (Eds.), Tutorials in Operations Research, INFORMS, 2008, pp. 261–269.

[27] H. Chernoff, A measure of asymptotic efficiency for tests of a hypothesis based on the sum of observations, The Annals of Mathematical Statistics 23 (1952).

[28] W. Hoeffding, Probability inequalities for sums of bounded random variables, Journal of the American Statistical Association 58 (1963) 13–30.

[29] A. Charnes, W. W. Cooper, Deterministic equivalents for optimizing and satisficing under chance constraints, Operations Research 11 (1963) 18–39.

[30] J. R. Birge, F. Louveaux, Introduction to Stochastic Programming, Springer Verlag, New York, 1997.

[31] T. Benoist, E. Bourreau, Y. Caseau, B. Rottembourg, Towards stochastic constraint programming: A study of online multi-choice knapsack with deadlines, in: T. Walsh (Ed.), Principles and Practice of Constraint Programming, CP 2001, Proceedings, volume 2239 of *LNCS*, Springer, 2001, pp. 61–76.

[32] T. Balafoutis, K. Stergiou, Algorithms for stochastic csps, in: F. Benhamou (Ed.), Principles and Practice of Constraint Programming, CP 2006, Proceedings, volume 4204 of *LNCS*, Springer, 2006, pp. 44–58.

[33] R. Rossi, S. A. Tarim, B. Hnich, S. D. Prestwich, Cost-based domain filtering for stochastic constraint programming, in: P. J. Stuckey (Ed.), Principles and Practice of Constraint Programming, 14th International Conference, CP 2008, Sydney, Australia, September 14-18, 2008. Proceedings, volume 5202 of *Lecture Notes in Computer Science*, Springer, 2008, pp. 235–250.

[34] J. C. Beck, N. Wilson, Proactive algorithms for job shop scheduling with probabilistic durations, J. Artif. Intell. Res. (JAIR) 28 (2007) 183–232.

[35] P. V. Hentenryck, R. Bent, Y. Vergados, Online stochastic reservation systems, in: J. C. Beck, B. M. Smith (Eds.), Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems, Third International Conference, CPAIOR 2006, Cork, Ireland, May 31 - June 2, 2006, Proceedings, volume 3990 of *Lecture Notes in Computer Science*, Springer, 2006, pp. 212–227.

[36] I. Katriel, C. Kenyon-Mathieu, E. Upfal, Commitment under uncertainty: Two-stage stochastic matching problems, in: L. Arge, C. Cachin, T. Jurdzinski, A. Tarlecki (Eds.), Automata, Languages and Programming, 34th International Colloquium, ICALP 2007, Wroclaw, Poland, July 9-13, 2007, Proceedings, volume 4596 of *Lecture Notes in Computer Science*, Springer, 2007, pp. 171–182.

[37] L. Michel, P. V. Hentenryck, Iterative relaxations for iterative flattening in cumulative scheduling, in: S. Zilberstein, J. Koehler, S. Koenig (Eds.), Proceedings of the Fourteenth International Conference on Automated Planning and Scheduling (ICAPS 2004), June 3-7 2004, Whistler, British Columbia, Canada, AAAI, 2004, pp. 200–208.

[38] R. Bent, P. V. Hentenryck, Regrets only! online stochastic optimization under time constraints, in: Proceedings of the Nineteenth National Conference on Artificial Intelligence, Sixteenth Conference on Innovative Applications of Artificial Intelligence, July 25-29, 2004, San Jose, California, USA, 2004, pp. 501–506.

[39] R. Bent, I. Katriel, P. V. Hentenryck, Sub-optimality approximations, in: P. van Beek (Ed.), Principles and Practice of Constraint Programming - CP 2005, 11th International Conference, CP 2005, Sitges, Spain, October 1-5, 2005, Proceedings, volume 3709 of *Lecture Notes in Computer Science*, Springer, 2005, pp. 122–136.

[40] D. S. Yates, D. S. Starnes, D. S. Moore, The practice of statistics, W H Freeman & Co, 2002.

[41] M. D. McKay, R. J. Beckman, W. J. Conover, A comparison of three methods for selecting values of input variables in the analysis of output from a computer code, Technometrics 21 (1979) 239–245.

[42] J. Neyman, On the problem of confidence limits, Annals of mathematical statistics 6 (1935) 111–116.

[43] F. Benhamou, L. Granvilliers, Continuous and interval constraints, in: F. Rossi, P. van Beek, T. Walsh (Eds.), Handbook of Constraint Programming, Elsevier, 2006, p. 569.